GUIDA PER L'APPLE

VOLUME 2

LE ESTENSIONI

Benoît de Merly

Guida per l'APPLE

2. Le estensioni

DELLO STESSO EDITORE

Volumi pubblicati

D.-J. David - La scoperta del Commodore 64
D.-J. David - La pratica del Commodore 64
J. Deconchat - 102 programmi per Commodore 64
X. Linant de Bellefonds - La pratica dello ZX Spectrum - Vol. 1
M. Henrot - La pratica dello ZX Spectrum - Vol. 2
J.-F. Sèhan - Chiavi per lo ZX Spectrum
J. Lévy - Esercizi per lo ZX Spectrum
J.-F. Sèhan - Alla ribalta: lo ZX Spectrum
C. Galais - Vademecum per Applesoft
J. Boisgontier - L'Apple e i suoi files
B. De Merly - Guida per l'Apple - Vol. 1
D. A. Lien - Dizionario del Basic
A. Pinaud - CP/M passo dopo passo.

Volumi di prossima pubblicazione

J. Boisgontier - Commodore 64: metodi pratici
J. Boisgontier, S. Brebion, G. Foucault - Il Commodore 64 per tutti
B. De Merly - Guida per l'Apple - Vol. 3
F. Lévy - Esercizi per l'Apple II
J. Boisgontier - 36 programmi per Apple IIe, II plus, IIc
J. Deconchat - 102 programmi per ZX Spectrum e ZX 81
J. Boisgontier - Il Basic per tutti

BENOÎT DE MERLY

Guida per l'APPLE

2. Le estensioni

Edizione italiana a cura di
FRANCO POTENZA

Traduzione di
ALBERT PALACI

Editsi - Editoriale per le scienze informatiche - S.r.l.
MILANO 1984

Sommario

CAPITOLO 1

Il sistema operativo: il DOS 3.3

1.1 PRESENTAZIONE GENERALE DEL DOS 3.3

Il DOS 3.3 è un sistema di gestione dei mini floppy (Disk Operating System) che permette di memorizzare delle informazioni su mini floppy e di ricaricarle in memoria quando lo si desidera.
Un mini floppy è un disco magnetico flessibile sul quale si possono memorizzare delle informazioni come su una cassetta. La differenza principale è che, prima di arrivare al dato voluto, bisogna srotolare tutta la cassetta, mentre con un mini floppy si ha un accesso molto più rapido.
Questa differenza si ritrova tra un disco musicale e una cassetta audio: potete accedere direttamente alla terza canzone del disco, mentre su una cassetta dovete ascoltare le due canzoni precedenti.
Inoltre un mini floppy permette di memorizzare più informazioni di una cassetta e di accedervi molto più rapidamente. Il DOS 3.3 permette di memorizzare, su un mini floppy, 140 K bytes di dati, ossia una quantità maggiore della capacità di memoria del vostro APPLE II.
Esistono diverse versioni del DOS. Il DOS 3.3 è l'ultima versione uscita per l'APPLE II.
In questo capitolo studieremo i comandi del DOS 3.3, le possibilità di gestione dei files di dati e di creazione di comandi, e faremo anche un accenno sulla struttura del DOS e delle informazioni sul mini floppy.
Gli esempi saranno per lo più scritti in APPLESOFT.

1.2 GENERALITÀ SULLA STRUTTURA DELLE INFORMAZIONI SUL MINI FLOPPY

I mini floppy sono dei dischi magnetici flessibili suddivisi in piste concentriche numerate, per il DOS 3.3, da 0 a 34. Una pista è a sua volta suddivisa in settori.

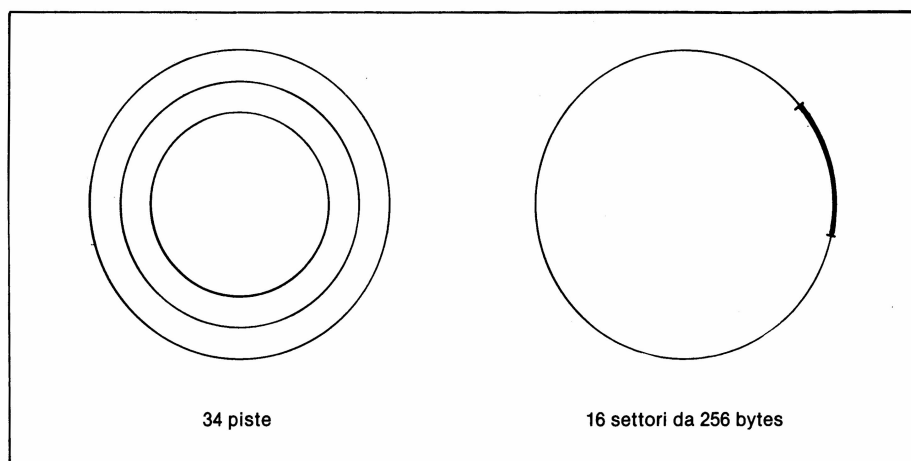



Foto J.

Nelle versioni precedenti del DOS, una pista conteneva 13 settori; nel DOS 3.3, invece, ci sono 16 settori per pista. Ogni settore contiene, in ogni caso, 256 bytes.
La busta del mini floppy permette l'accesso con la testina di lettura, tramite un intaglio radiale. La testina di lettura, per accedere alle piste, si sposta.

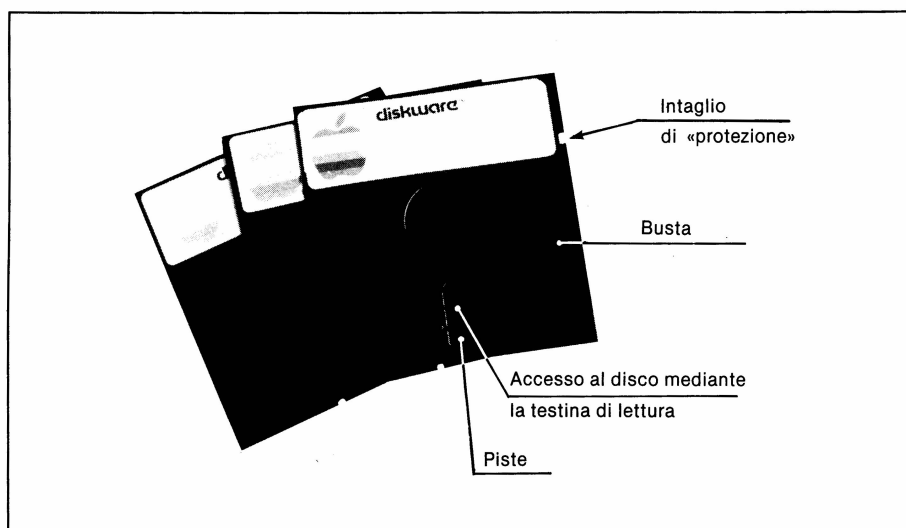



Foto K.

L'accesso ai settori si ottiene facendo girare il mini floppy all'interno della sua busta.
L'intaglio di protezione permette la scrittura sul mini floppy; coprendolo si vieta la scrittura su di esso.

1.3 CARICAMENTO DEL DOS 3.3

Per accedere ai comandi del DOS è necessario averlo prima caricato in memoria. Ci sono vari modi per eseguire il caricamento; la scelta di uno di questi dipende dalla versione dell'APPLE II e dal linguaggio in cui vi trovate.
In ogni caso supporremo che il mini floppy "SYSTEM MASTER" sia nel drive 1 dell'unità di controllo situata nello slot 6.

1.3.1 ROM autostart. APPLE II + e APPLE II e.

È il metodo più semplice. È sufficiente accendere il vostro APPLE II. Quando il DOS è stato caricato compare il carattere].

1.3.2 Caricamento da monitor

Quando vedete un asterisco sullo schermo significa che il monitor è in attesa di un vostro comando. Avete allora tre possibilità:

*C600 G	salto all'indirizzo $C600 dell'inizio del programma di caricamento del DOS
*6 Ctrl-K	(vedere monitor dell'APPLE)
*6 Ctrl-P	

Il DOS a questo punto è stato caricato.

1.3.3 Caricamento da Basic

Lo stesso comando è usato per i due tipi di Basic. Potete digitare uno dei comandi seguenti:

PR#6
IN#6

1.3.4 Caricamento di una vecchia versione del DOS

In questo caso il mini floppy Basic deve essere inserito al posto del mini floppy "SYSTEM MASTER". Lanciate ora il caricamento del DOS. Quando compare il messaggio:

INSERT BASIC DISK AND PRESS RETURN

oppure

INSERT 13 SECTORS DISK AND PRESS RETURN

mettete il mini floppy del vecchio sistema. A questo punto inizia il caricamento della vecchia versione del DOS.

1.4 I COMANDI GENERALI DEL DOS 3.3

1.4.1 Opzioni generali

La maggior parte dei comandi del DOS 3.3 vi consentono di specificare alcune opzioni: scelta di un'unità disco tra le due gestite dall'unità di controllo, scelta dello slot contenente l'unità di controllo da usare, oppure scelta del volume del mini floppy.
Il formato di queste tre opzioni è sempre uguale e questa è la ragione per cui li studiamo prima degli altri comandi. Queste opzioni modificano i valori presi per "default" dal DOS quando i comandi non specificano le opzioni.
Per esempio, se in un comando specifichiamo l'opzione S5 (scelta dello slot 5), verrà utilizzata l'unità di controllo situata nello slot 5 per tutti i comandi del DOS, finché non viene richiamata nuovamente l'opzione S.
Queste tre opzioni hanno il seguente formato:

Ss dove s è il numero dello slot (da 1 a 7)
Dd dove d è il numero del drive (da 1 a 2)
Vv dove v è il volume del mini floppy (da 1 a 254)

Il numero di drive corrisponde alla posizione occupata dal cavo che collega il drive all'unità di controllo.
Il volume di un mini floppy è un parametro che, sull'APPLE II, non ha significato fisico. Il DOS 3.3 usa questo dato esclusivamente in un comando, per poterlo paragonare con il volume del mini floppy corrente. Se c'è una differenza compare il messaggio d'errore VOLUME MISMATCH e il DOS non esegue il comando.
Se nel vostro comando non specificate il volume, il DOS ignora quest'ultimo.
Il volume, essendo indicato nel comando CATALOG (cfr. par. 5.4.3), può essere usato per numerare i vostri mini floppy oppure per distinguere i dischi su cui avete salvato i vostri programmi.
Il volume vi servirà ogni volta che vorrete distinguere dei mini floppy allo scopo di proteggerli.

Esempio: CATALOG D2, S5, V1

Se in un comando viene dato un numero di slot (o un numero di drive) errato, il DOS farà comparire:

I/O error

ed emetterà un "beep".
Per recuperare il vostro programma, dovrete seguire le seguenti istruzioni:

1. CATALOG, Ss′ dove s′ è il numero corretto di slot.
2. Quando il sistema si ferma, premere il tasto Reset.
3. Digitare, se necessario (cioè nel caso in cui vi ritroviate nel monitor), 3DOG.

1. *Il* DOS *dev'essere caricato sempre dall'unità disco* 1.
2. *Non dovete mai premere il tasto Reset quando il drive è in funzione (spia rossa accesa): il mini floppy inserito andrebbe perso.*

1.4.2 Richiesta del catalogo di un mini floppy

Il comando CATALOG del DOS 3.3 permettte di ottenere l'elenco dei files contenuti in un mini floppy.
Questo comando visualizza per prima cosa il volume del mini floppy; poi indica, per ogni file, se esso è bloccato o no, il tipo dei dati del file, il numero di settori di 256 bytes occupati dal file e il nome del file.
Un mini floppy DOS 3.3 può contenere fino a 84 files (limitazione della dimensione del catalogo).
Un file bloccato è un file protetto in scrittura. Dunque non può essere né cancellato né distrutto. I files bloccati sono indicati nel catalogo da un asterisco che precede il tipo del file.
I 4 tipi di file sono rappresentati da una lettera:

T file di testo
I programma Integer Basic
A programma APPLESOFT
B codice binario

La lunghezza del file è il numero di settori occupati dal file stesso. Se il file supera 255 settori, la lunghezza segnalata dal CATALOG è la lunghezza reale modulo 256.
Il nome di un file ha una lunghezza compresa tra 1 e 30 caratteri. Deve iniziare con una lettera e può contenere tutti i caratteri tranne la virgola. Potete mettere nel nome di un file dei caratteri di controllo, ma questi non compaiono sullo schermo. Questo è un modo di impedire ad altre persone di utilizzare i vostri files, ma fate attenzione a non dimenticarvi i caratteri che avete digitato!
Il formato del comando CATALOG è il seguente:

CATALOG Dd, Ss, Vv

Esempio: Se il numero totale di linee supera le 20, il DOS visualizzerà le prime 21 linee e aspetterà che battiate un qualsiasi tasto (tranne Reset, CTRL e SHIFT) per visualizzare la fine del catalogo. Questo vi dà il tempo necessario per leggere il nome di tutti i files.

```
] CATALOG
DISK VOLUME 254
*A 006 HELLO
```

*I 051 BRIDGE1
*I 007 CLOCK 2
*I 045 YAM'S
*A 009 B.NAVALE
*A 005 LUNARLANDING
*A 013 RADIO AMATEUR
*A 030 DYNASTY
*A 036 BOWLING
*A 029 JEUX MATHS
*I 025 CRYPTOGRAM
*I 015 COWBOY
*B 003 SHOOTOUT
*I 029 LOGIC DESIGNER
*I 025 COLOR PHOTO
*I 029 PHOTO NEG FILE
*A 031 WORLD POWER 6
*I 013 RAID

1.4.3 Inizializzazione di un mini floppy

Il mini floppy "System master", fornito con il DOS, è un mini floppy pronto per l'uso. Non sarà così per gli altri mini floppy in commercio.
Per convincervi di questo, estraete il vostro mini floppy "System master" e sostituitelo con un mini floppy nuovo. Provate ora a ricaricare il vostro APPLE.
Digitate, per esempio, PR#6. Cosa succede? L'unità disco si mette in funzione, la spia rossa si accende, si sentono strani rumori, il mini floppy inizia a girare e non si ferma.
Dovete, allora, premere il RESET. Vi avevamo detto, in precedenza, di non premere mai il Reset per non correre il rischio di perdere il contenuto del mini floppy. In questo caso, dato che il mini floppy è vuoto, non perderete niente.
Cos'è successo? L'APPLE ha cercato sul mini floppy delle informazioni non esistenti. Un mini floppy vergine è come una cassetta vergine. È vuoto. Prima di utilizzarlo dovete inizializzarlo.
Il comando INIT permette di inizializzare i mini floppy "secondari". I mini floppy secondari dipendono dalla dimensione della memoria del sistema. La dimensione della memoria usata dal mini floppy è la stessa di quella del sistema che ha inizializzato il mini floppy.
Se un mini floppy viene creato con una memoria di 16 K bytes, anche se usiamo un APPLE II da 32, 48 o 64 K bytes, saranno usati solo i primi 16 K bytes.
È possibile trasformare un mini floppy secondario in un mini floppy "master" usando il programma di utility MASTER CREATE (cf. par. 1.8). Il DOS, così, si carica automaticamente ad un indirizzo che permetterà un uso più efficiente della memoria.

Il formato del comando INIT è il seguente:

INIT nome del file, Ss, Dd, Vv

Questo comando assegna il volume al mini floppy. Se non specificate il volume o se assegnate a v il valore 0, il volume viene considerato, per default, 254.
Perché si deve precisare il nome del file nel comando INIT? La ragione è semplice. Quando caricate il DOS da un mini floppy, questo metterà in memoria il file il cui nome avrete indicato nel comando INIT e lancerà l'esecuzione del programma corrispondente.

1. *Il file specificato nel comando* INIT *viene creato da questo comando copiando il programma Basic in memoria sul mini floppy.*
2. *Le informazioni situate sul mini floppy vengono cancellate dal comando* INIT. *Fate dunque molta attenzione a non inizializzare dei mini floppy non vuoti, a meno che non vogliate recuperare un vecchio mini floppy.*
3. *Vi consigliamo di utilizzare sempre lo stesso nome per il file da lanciare al momento del caricamento. Per esempio* "BUONGIORNO". *Inoltre sarebbe utile che questo programma indicasse le caratteristiche del mini floppy.*

Eccovi un esempio in cui viene utilizzato il comando INIT con l'APPLESOFT:

1. Estraete il mini floppy "System Master" e sostituitelo col mini floppy da inizializzare.
2. Scrivete il programma che deve essere eseguito ogni volta che si carica il DOS. Eccovi un esempio di un programma di questo tipo.
3. Digitate le seguenti istruzioni:

```
]NEW
]10 PRINT "MINI FLOPPY SECONDARIO 48K"
]20 PRINT "CREATO IL 5 MAGGIO 1982"
]30 END
]INIT HELLO
I/O ERROR                      non ci sono floppy nel drive 1
]INIT HELLO, D2                inizializzazione mini floppy
]CATALOG, D2              |
DISK VOLUME 254           }    verifica
A 002 HELLO               |
]RUN
MINI FLOPPY SECONDARIO 48K  }
CREATO IL 5 MAGGIO 1982     }  prova di HELLO
```

4. Quando il sistema si ferma, ricaricate il DOS. Vedrete apparire il commento:

APPLE II
MINI FLOPPY SECONDARIO 48K
CREATO IL 5 MAGGIO 1982

5. Mettete da parte il mini floppy "System Master". Il mini floppy che avete appena inizializzato vi servirà per provare gli esempi dei paragrafi seguenti.

Avete potuto verificare, seguendo questo esempio, che il DOS ha inizializzato il mini floppy contenuto nel drive 2 che, da questo momento, è accessibile all'utente e che contiene il programma HELLO.

1.4.4 I files bloccati

Se volete proteggere un programma da una possibile distruzione, usate il comando LOCK del DOS che bloccherà il file. Il formato di questo comando è il seguente:

LOCK nome file, Ss, Dd, Vv

A questo punto se provate a cancellare il file oppure a cambiargli nome, il DOS rifiuterà il vostro comando e darà il seguente messaggio:

FILE LOCKED (file bloccato)

Se tentate di salvare sul mini floppy (cfr. par. 1.5.1) un programma scritto nello stesso linguaggio del file bloccato, otterrete lo stesso risultato. Invece, se volete salvare con lo stesso nome un file di tipo differente, vedrete comparire il messaggio:

FILE TYPE MISMATCH (tipo del file inappropriato)

In un catalogo i files bloccati si distinguono dagli altri per un asterisco (cfr. par. 1.4.2).
Vi raccomandiamo di proteggere il file eseguito al caricamento del DOS (par. 1.4.3).
È possibile sbloccare un file. Dovrete usare il comando UNLOCK il cui formato è il seguente:

UNLOCK nome file Ss, Dd, Vv

Il file non sarà più protetto e potrà essere distrutto o cancellato da una nuova versione del programma.

Esempio:

```
]LOCK HELLO
]CATALOG

DISK VOLUME 254

*A 002 HELLO
```

]UNLOCK HELLO
]CATALOG

DISK VOLUME 254

A 002 HELLO

1.4.5 Distruzione di un file

Il comando DELETE permette di cancellare un file. Il formato del comando è il seguente:

DELETE nome file, Ss, Dd, Vv

Per esempio le istruzioni:

]LOAD HELLO
]SAVE BUONGIORNO
]DELETE BUONGIORNO
]CATALOG

DISK VOLUME 254

*A 002 HELLO

creeranno il file BUONGIORNO e poi lo elimineranno dal mini floppy situato nel drive attivo.
Se il file da cancellare è bloccato, il DOS dà il messaggio d'errore FILE LOCKED e ignora il comando.

Esempio:

LOCK HELLO
]DELETE HELLO

FILE LOCKED

1.4.6 Cambio del nome di un file

Il comando RENAME permette di modificare il nome di un file. Il suo formato è il seguente:

RENAME nome da cambiare, nuovo nome

Come per il comando DELETE, se il file è bloccato, il DOS ignora il comando e dà il messaggio FILE LOCKED.

Il DOS *non controlla se esiste già un file il cui nome è quello che si vuole dare al file da ridenominare e sostituisce, nel catalogo, il vecchio nome col nuovo. Avrete così, sul mini floppy, due files con lo stesso nome. Questo potrà provocare numerosi equivoci. Fate dunque molta attenzione a questo fenomeno.*

1.4.7 Verifica fisica di un file

Il comando VERIFY consente di verificare che i settori occupati da un file non siano danneggiati fisicamente. Se il DOS trova un errore manda il messaggio:

I/O ERROR

Altrimenti non compare alcun messaggio. Il formato di questo comando è il seguente:

VERIFY nome file, Ss, Dd, Vv

Potete usare questo comando per tutti i tipi di file, dai due tipi di Basic e da monitor.

1.4.8 Passaggio da un Basic all'altro

È realizzato tramite i comandi:

FP (APPLESOFT) del DOS
INT (Integer Basic)

Si possono presentare diversi casi:

- siete in Basic APPLESOFT e battete FP. Questo distrugge il vostro programma e i vostri dati e restate in Basic APPLESOFT;
- la situazione è identica per INT e l'Integer Basic;
- siete in un Basic e volete passare all'altro:
  - digitate FP e vi ritroverete in APPLESOFT;
  - digitate INT e vi ritroverete in Integer Basic.

Questa possibilità viene offerta solo agli utenti che possiedono o la scheda dei linguaggi o l'APPLE IIe; infatti questo consente di avere un Basic in ROM *e l'altro nella RAM della scheda dei linguaggi (files* INTBASIC *e* FPBASIC).
Per gli altri utenti, l'unica soluzione è di caricare l'altro Basic da una cassetta o da un mini floppy in RAM; questo però porta ad una diminuzione della memoria disponibile.

Esempio:

```
LOAD HELLO
]LIST
10 PRINT " MINI FLOPPY SECONDARIO 48K"
20 PRINT " CREATO IL 5 MAGGIO 1982"
30 END
]INT
>LIST
```

```
>10 PRINT "COUCOU"
>20 END
>LIST

    10 PRINT "COUCOU"
    20 END

>FP

]LIST
```

1.4.9 Comandi per la messa a punto del DOS, MON, TRACE, NOMON

• MON, NOMON

I comandi MON, NOMON del DOS consentono di ottenere le tracce visive dei comandi e dei dati relativi ai files manipolati dal DOS. Esistono tre opzioni:

Commands (comandi del DOS OPEN, CLOSE...)
Inputs (dati letti dal mini floppy)
Outputs (dati scritti sul mini floppy)

Il formato dei comandi è

```
   MON C,I,O    attivazione delle tracce
 NOMON C,I,O    disattivazione delle tracce
```

Per default, è in atto il NOMON C,I,O (nessuna traccia).
Il comando MON rimane attivato finché non si incontra NOMON, INT, FP, Reset, 3DOG.
L'esempio seguente chiarirà l'uso dei comando MON e NOMON con le opzioni C, I oppure O.

```
]MON C
]PRINT "CLOSE"      chiusura di tutti i files
CLOSE               stampato da MON

]
]NOMON C, I, O
]PRINT "CLOSE"
]
```

• TRACE (APPLESOFT)

Il comando Trace dell'APPLESOFT non funziona col DOS; eviteremo dunque di usarlo.

1.5 I FILES DI PROGRAMMI

1.5.1 Programmazione in Basic

In questo paragrafo vi spiegheremo come usare il DOS quando dovete salvare e caricare i vostri programmi Basic. Saprete inoltre usare il DOS anche all'interno dei vostri programmi Basic.

• Caricamento di un programma Basic in memoria

Il caricamento di un programma in memoria si esegue tramite il comando LOAD, il cui formato è il seguente:

LOAD nome file, Vv, Dd, Ss

La differenza dal caricamento da cassetta consiste nel fatto che è necessario specificare il nome del file e che il caricamento è molto più rapido.
Per esempio, se volete caricare in memoria il programma APPLESOFT chiamato COLOR DEMOSOFT dovrete usare l'istruzione LOAD COLOR DEMOSOFT.
Il DOS cerca il file sul mini floppy, cancella qualsiasi programma Basic che si trova in memoria e carica il file. Se avete usato il nome di un file non conosciuto dal DOS, esso farà comparire il messaggio d'errore FILE NOT FOUND.
Quando il DOS è pronto per ricevere un comando, potete visualizzare, modificare o eseguire il vostro programma. Per eseguire un programma è inutile digitare LOAD programma e RUN, come si fa su cassetta; esiste infatti un comando del DOS che unisce le due istruzioni.

• Esecuzione di un programma Basic salvato sul mini floppy

Il comando RUN del DOS permette di unire in un'unica istruzione i comandi LOAD del DOS e RUN del Basic. Il suo formato è il seguente:

RUN file, Dd, Vv, Ss

Esempio:

```
]LOAD HELLO
]LOAD HELLO, D2
]RUN
MINI FLOPPY SECONDARIO 48K
CREATO IL 5 MAGGIO 1982

]RUN HELLO, D2
MINI FLOPPY SECONDARIO 48K
CREATO IL 5 MAGGIO 1982
```

Il DOS cerca il file sul mini floppy, lo carica in memoria ed esegue il programma corrispondente. Se non trova il file dà il messaggio d'errore:

FILE NOT FOUND

e si mette in attesa di nuovi comandi.

• Memorizzazione di un programma Basic sul mini floppy

Sul mini floppy "SYSTEM MASTER", oltre al file HELLO da eseguire, avete altri files. Questi sono stati memorizzati sul mini floppy col comando SAVE del DOS. Anche voi potete memorizzare i vostri programmi sui vostri mini floppy. Consideriamo ad esempio il programma:

```
 5 HOME
10 PRINT "COUCOU"
20 END
```

Mettete un mini floppy inizializzato nel drive uno dell'unità di controllo situata nello slot 6 e battete

SAVE COUCOU, D1, S6.

Cosa succede? Il DOS salva il vostro programma Basic, che si trova in memoria, sul mini floppy. Per esserne sicuri, al prompt, digitate CATALOG; il file COUCOU è presente sul mini floppy. Potete anche spegnere il vostro APPLE, tornare al Basic, caricare il DOS e ricaricare il vostro programma (LOAD COUCOU). Ed eccovi lo stesso programma. Visualizzatelo! Otterrete il seguente risultato:

```
 5 HOME
10 PRINT "COUCOU"
20 END
```

Il comando SAVE ha memorizzato correttamente il vostro programma sul mini floppy.
Se date al vostro programma un nome che esiste già su un altro file, il DOS cancellerà il vecchio file per sostituirlo con la nuova versione. Se per sbaglio i due files sono di diverso tipo, il DOS rifiuterà il comando e darà il seguente messaggio d'errore.

FILE TYPE MISMATCH

Se il file che volete cancellare è bloccato (cfr. par. 1.4.4), il DOS rifiuterà ancora il comando e darà il messaggio d'errore:

FILE LOCKED

Il formato del comando SAVE è il seguente:

SAVE nome file, Vv, Ss, Dd

Dato che il comando SAVE *non modifica il programma in memoria, potete utilizzare il comando Verify per essere sicuri che i settori usati nella memorizzazione non siano danneggiati fisicamente.*

Esempio:

]SAVE BUONGIORNO

]CATALOG

DISK VOLUME 254

*A 002 HELLO
 A 002 BUONGIORNO

• **Uso dei comandi del DOS in un programma Basic**

Potete usare comandi del DOS nei vostri programmi Basic. Alcuni di questi comandi possono essere usati esclusivamente in programmi Basic: sono i comandi che manipolano dati (cfr. par. 1.6: I files di dati).
Per usare un comando del DOS in un programma Basic dovete utilizzare l'istruzione PRINT e far precedere il comando dal codice ASCII 4 (CTRL-D). Se volete ad esempio visualizzare il catalogo sullo schermo dovete usare l'istruzione:

PRINT "CATALOG"
il CTRL-D è stato digitato prima della C del comando CATALOG

Siccome il carattere CTRL-D non è visibile, normalmente al suo posto si usa la variabile D$.
Con l'APPLESOFT dovrete inserire all'inizio del programma:

D$ = CHR$(4):REM D$ = CTRL-D

L'istruzione che avevamo precedentemente scritto diventa:

PRINT D$; "CATALOG"
CTRL = D senza CTRL-D prima di CATALOG

I vostri programmi diventano così più leggibili e più facilmente correggibili.

1.5.2 Files binari - programmi assembler

Il DOS consente di salvare sul vostro mini floppy e di caricare in memoria dal mini floppy le informazioni contenute nella memoria dell'APPLE quali sottoprogrammi assembler o forme grafiche predefinite. Vi abbiamo spiegato i comandi LOAD, SAVE e RUN dei programmi Basic; i comandi BLOAD, BSAVE e BRUN, trattati in questo paragrafo, hanno le stesse funzioni con zone di memoria designate dai loro indirizzi e dalle loro lunghezze.
Il carattere B, all'inizio di questi tre comandi, significa file binario. Questi files sono la rappresentazione esatta di una zona di memoria.

• **Memorizzazione di una zona di memoria su mini floppy**

Il comando BSAVE del DOS consente di copiare in un file binario una zona di memoria. Il formato di questo comando è il seguente:

BSAVE file, Aa, Ll, Dd, Vv, Ss

I parametri A e L non sono facoltativi.
Il parametro A rappresenta l'indirizzo d'inizio del segmento di memoria da copiare nel file. Questo indirizzo può essere in forma decimale o esadecimale. Nel primo caso, deve essere compreso tra i valori 0 e 65535. I numeri negativi, usati in Basic per rappresentare gli indirizzi alti della memoria, non sono ammessi. Nel secondo caso, l'indirizzo deve essere preceduto dal simbolo $ e dev'essere compreso tra $0000 e $FFFF. Se l'indirizzo specificato non si trova in questo intervallo, il DOS emetterà il messaggio SYNTAX ERROR e ignorerà il comando. Se, invece, l'indirizzo non corrisponde fisicamente ad un chip (a causa della dimensione del sistema), il DOS non darà messaggi ma non considererà il comando.
Il parametro L rappresenta la lunghezza della zona di memoria in questione. Anch'esso può essere espresso in forma decimale o esadecimale. Se la lunghezza non è compresa tra 0 e 65535, il DOS darà il messaggio SYNTAX ERROR e ignorerà il comando. Infatti la lunghezza deve essere inferiore a 32 K bytes, altrimenti il monitor presenterà il messaggio RANGE ERROR e non considererà il comando. Se volete salvare una zona di memoria superiore a 32 K bytes dovete usare due volte il comando BSAVE. Come per il parametro A, il DOS non verifica la validità fisica della lunghezza.
Se volete, ad esempio, memorizzare una pagina grafica ad alta risoluzione (in questo caso la seconda), potete digitare uno dei comandi:

```
BSAVE FHGR2, A$4000, L$2000, D1, S6
BSAVE FHGR2, A$4000, L8192, V254, D2
BSAVE FHGR2, A16384, L$2000, S5
BSAVE FHGR2, A16384, L8192
```

• Caricamento di una zona di memoria da un mini floppy

Il comando BLOAD del DOS consente di caricare una zona di memoria da un file sul mini floppy. Se digitate, ad esempio:

```
BLOAD FHGR2 (cf. qui sopra)
POKE-16304,0
POKE-16299,0
```

ritrovate sullo schermo la pagina grafica che avevate memorizzato.
Il formato del comando è il seguente:

BLOAD nome file Aa, Ss, Dd, Vv

Il parametro A rappresenta, come per BSAVE, un indirizzo; ma questa volta si tratta dell'indirizzo di caricamento. Specificate questo parametro solo se volete caricare la zona di memoria ad un indirizzo diverso da quello indicato nel comando BSAVE.
A differenza del comando LOAD, che cancella qualunque programma BASIC in memoria, il comando BLOAD cancella solo la zona sulla quale applica il file.

1. *Siccome la lunghezza della zona è determinata dalla lunghezza del file, non è necessario precisare la lunghezza nel comando* BLOAD.
2. *Un programma in linguaggio macchina o in assembler può funzionare solo se viene caricato ad un dato indirizzo: l'indirizzo a partire dal quale era stato memorizzato. Generalmente utilizzerete solo il comando* BLOAD *nome file.*

• **Esecuzione di un programma in linguaggio macchina da un mini floppy**

Il comando BRUN esegue il comando BLOAD e un'istruzione di salto all'indirizzo d'inizio della zona di memoria.

BRUN nome file, Aa, Ss, Dd, Vv

Il parametro A è uguale a quello del comando BLOAD. Se il file contiene un programma in assembler, BRUN carica e lancia l'esecuzione del programma.

1.6 I FILES DI DATI

1.6.1 Generalità

I files di dati sono del tipo testo T. Sono indicati nel catalogo dalla lettera T presente nella colonna del tipo.
I files di dati possono essere creati da un programma in un dato linguaggio e riletti in un altro linguaggio. Per creare o utilizzare dei files di dati, dovrete chiamare i comandi del DOS da un programma Basic (cfr. par 1.5.1).
I seguenti comandi non possono essere usati con files di dati:

LOAD, RUN, SAVE, BLOAD, BRUN, BSAVE

Per memorizzare e richiamare dei dati in un file di dati, dovete scrivere dei programmi Basic. Per questo motivo, sono stati creati i seguenti comandi del DOS

OPEN
CLOSE
READ
WRITE
APPEND
POSITION
EXEC

Tra questi comandi, solo CLOSE e EXEC sono accessibili direttamente dall'utente.
Per i files di dati potete usare anche i comandi:

LOCK, UNLOCK, DELETE, RENAME, VERIFY, CATALOG, MON, NOMON
(cfr. *Comandi generali del* DOS)

Il DOS 3.3 gestisce due tipi di file di dati:

- i files sequenziali, nei quali i dati sono registrati uno dopo l'altro e sono letti allo stesso modo (cfr. *Files su cassetta*);
- i files ad accesso diretto, più complicati da gestire, ma che danno migliori prestazioni, e nei quali si può leggere o scrivere un record in qualunque punto del file, senza dover percorrere i records precedenti.

Studieremo, in seguito, questi due tipi di file spiegando come usarli e le differenze tra APPLESOFT e Integer Basic.

1.6.2 Il comando MAXFILES

Il DOS consente, per default, di manipolare simultaneamente tre files. Se questo non vi basta, potrete, dopo aver utilizzato il comando:

MAXFILES n (dove n = numero dei files)

lavorare, al massimo, su 16 files.
Quando il comando è stato eseguito, il DOS sposta HIMEM (massimo indirizzo di memoria utilizzabile per il Basic) verso il basso, in modo da riservare, per le operazioni di input/output, un buffer di 595 bytes per file. Questo può distruggere: in APPLESOFT, la zona di memoria occupata dalle stringhe di caratteri e, in Integer Basic, il programma dell'utente. Per evitare questo fatto, vi consigliamo di:

- specificare sempre il numero di files necessari prima del caricamento del programma;
- caricare ed eseguire il vostro programma.

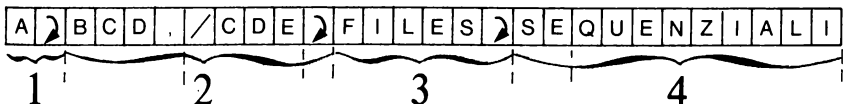
Un'altra soluzione:

```
 5 PRINT "CTRL-D MAXFILES N":END
10 REM INIZIO DEL PROGRAMMA
```

- eseguire il programma (MAXFILES = N);
- ricaricarlo;
- saltare alla linea 10 (GOTO 10).

1.6.3 I files ad accesso sequenziale

• **Relazione tra settori fisici e campi**

I campi sono costituiti da stringhe di caratteri di lunghezza variabile separati dai codici di ritorno (↵). I vari dati di una stringa sono separati da virgole.

La gestione dei campi logici è indipendente, per l'utente, dalla gestione dei settori fisici; è il DOS che effettua il collegamento logico-fisico.
Durante le operazioni di input-output su un file sequenziale, vengono eseguiti degli scambi di stringhe di caratteri tra il vostro programma e il file sequenziale. I campi sono separati tra loro dal carattere Return.

• Apertura di un file sequenziale

I file sequenziali, prima di essere usati, devono essere aperti. L'apertura di un file porta il DOS a cercare il file sul mini floppy, a inizializzare i dati di gestione del file e a riservare un buffer di 256 bytes per caricare in memoria un settore fisico, a evitare di dover accedere al disco per ogni campo e, dunque, risparmiare molto tempo.
Il formato del comando per l'apertura di un file sequenziale è il seguente:

OPEN nome file Ss, Vv, Dd

Se il file non esiste sul mini floppy, il DOS lo crea e lo apre.
Consideriamo il seguente programma:

```
1000 D$=CHR$(4):PRINT D$; "OPEN FSEQ":END
```

Digitate RUN. Il DOS aprirà il file FSEQ. Digitate CATALOG. Vedrete che il file FSEQ esiste:

T 001 FSEQ

• Chiusura di un file sequenziale

Quando avete finito di lavorare su un file sequenziale, dovete chiuderlo per memorizzare sul mini floppy le modifiche apportate al file e liberare il buffer usato per gli input-output sul file. Inoltre, se cambiate il mini floppy prima di aver chiuso i files, rischiate di avere dei problemi molto gravi sui mini floppy in questione. Alla fine dei vostri programmi, ricordatevi sempre di chiudere i vostri files sequenziali.
Il comando CLOSE permette di chiudere i files sequenziali. Questo comando ammette due formati:

CLOSE che chiude tutti i files aperti su tutti i mini floppy presenti
CLOSE file Vv, Ss, Dd che chiude il file sequenziale specificato.

• Scrittura di un file sequenziale

La scrittura dei dati si effettua usando l'istruzione PRINT, come sulla stampante o sullo schermo. Potete scrivere in un file sequenziale tutto ciò che visualizzate sullo schermo. Quando scrivete in un file sequenziale, il DOS sposta un puntatore al primo posto disponibile nel file, esattamente come una stampante fa avanzare la carta. Al momento dell'apertura del file questo puntatore era stato messo a zero; è stato dunque, inizializzato.
A questo punto dovete porvi una domanda. Come si può redirigere la scrittura dei dati verso il file invece che verso lo schermo? Il comando WRITE del DOS lo consente.

Se in un programma avete la linea PRINT D$; “WRITE FSEQ” con D$ = CTRL–D, tutte le istruzioni PRINT incontrate nel resto del programma scriveranno nel file sequenziale FSEQ.

Esempio:

```
10 D$ = CHR$(4)
20 PRINT D$; “OPEN FILESEQ, D2, S6”
30 PRINT D$; “WRITE FILESEQ”
40 PRINT “BUONGIORNO”
50 PRINT D$; “CLOSE FILESEQ”
60 END
```

Questo programma scrive BUONGIORNO nel primo campo del file FILESEQ. Per poterlo fare effettua le seguenti operazioni:

1. Apertura del file FILESEQ.
2. Redirezione dei dati nel DOS e nel file.
3. Scrittura del campo BUONGIORNO nel file.
4. Chiusura del file per evitare di perdere i dati.

Potete mettere in un campo quante stringhe di caratteri volete. Se sostituite la linea 40 con:

```
40 PRINT “BUONGIORNO”, “OK”
```

oppure

```
40 PRINT “BUONGIORNO”; :PRINT “OK”
```

scrivete due dati nel file. Infatti, finché l’istruzione PRINT non dà Return, il DOS non considera finito il campo. Potete anche sostituire la linea 40 con:

```
40 PRINT A$; B$; C$
```

Il contenuto delle tre variabili sarà registrato nello stesso campo del file.
La separazione dei dati è effettuata dalle virgole. Mentre il punto e virgola concatena i dati scritti in modo da crearne uno solo nel file, la virgola memorizza i dati senza concatenarli.
Tuttavia, una virgola messa alla fine di un’istruzione PRINT sopprime l’invio del Return, esattamente come il punto e virgola.
Se sostituite la linea 40 con la linea:

```
40 PRINT A$:PRINT C$
```

avrete tre campi nel vostro file.
La scrittura di dati in un file si effettua usando l’istruzione PRINT, senza cambiare il suo formato dopo la redirezione dei dati, e il comando WRITE del DOS. Quando avete finito di scrivere i dati nel file vi serve ritornare alla visualizzazione sullo schermo. Potete farlo in tre modi:

- usando un comando del DOS;
- inviando al DOS un comando vuoto

```
PRINT D$    con D$ = CTRL–D
```

- usando l'istruzione INPUT nel programma. In questo caso, tutti i caratteri da visualizzare prima dell'uso dei dati, compreso il punto di domanda, vengono scritti nel file.

I dati non sono più scritti nel file, ma sullo schermo o sulla stampante.
La visualizzazione di un messaggio d'errore cancella l'effetto del comando WRITE *ma solo dopo la scrittura dell'intero messaggio d'errore.*
Consideriamo il seguente programma:

```
 10 REM CREAZIONE DI UN FILE SEQUENZIALE
 20 D$ = CHR$ (4)
 30 HOME
 40 PRINT "   QUESTO PROGRAMMA VI CONSENTE DI CREARE" ;
 50 PRINT "UN FILE SEQUENZIALE"
 60 PRINT "ALLO SCOPO INSERITE UNA STRINGA DI " ;
 70 PRINT "CARATTERI DI LUNGHEZZA COMPRESA TRA 1" ;
 80 PRINT "E 239 CARATTERI"
 90 PRINT
100 PRINT "BATTETE RETURN PER FERMARVI"
110 PRINT D$ ; "OPEN FSEQ"
120 HOME : I = 1
130 PRINT "INSERITE LA STRINGA NUMERO "; I ; " : " ;
140 INPUT ""; A$
150 IF A$ = "" GOTO 210
160 PRINT D$ ; "WRITE FSEQ"
170 PRINT A$
180 PRINT D$
190 I = I + 1
200 GOTO 130
210 PRINT D$ ; "CLOSE"
220 END
```

Provatelo! Questo programma apre il file sequenziale FSEQ e scrive le stringhe di caratteri prese da tastiera, a partire dall'inizio del file.
Supponiamo che digitiate la prima volta le stringhe BUONGIORNO e OK, otterrete il file:

BUONGIORNO ⤸ OK ⤸

Lanciate nuovamente il programma e digitate, ad esempio, solo la stringa OK. Otterete il file:

OK ⤸ NGIORNO ⤸ OK ⤸

Cosa notate? La parte iniziale del file è stata annullata e sostituita con OK. Ma le informazioni seguenti non sono state cancellate. Infatti quando aprite un file, il DOS mette il puntatore all'inizio del file, ma non distrugge le informazioni precedentemente scritte. È dunque normale conservare, in alcuni casi, una parte dei vecchi dati.

Cosa fare per evitare questo fenomeno? Ci sono due possibilità:

- Desiderate conservare i vecchi dati e vi serve posizionare il puntatore alla fine del file (cfr. comandi APPEND e POSITION).
- Non volete conservare i vecchi dati: la miglior soluzione è quella di distruggere il file usando il comando DELETE. Si pone, però, un nuovo problema: se il file, al momento dell'esecuzione del comando DELETE, non esiste, il DOS manda un errore e il programma si ferma. Una soluzione molto semplice consiste nell'usare il comando OPEN col comando DELETE. In questo modo, il file esistente viene aperto e il file non esistente viene creato. Il comando DELETE, dunque, troverà sempre il file da distruggere.

Per aggiungere questa soluzione al programma precedente, è sufficiente aggiungere le seguenti due linee:

```
108 PRINT D$; "OPEN FSEQ"
109 PRINT D$; "DELETE FSEQ"
```

In questo modo, non avrete mai sovrapposizioni di dati tra due versioni di un file sequenziale.

• **Lettura dei dati di un file sequenziale**

I dati possono essere letti da un file nella stessa maniera in cui essi sono rediretti verso un file. Il comando READ file avverte il DOS che i dati non saranno più presi da tastiera, ma semplicemente letti nel file specificato.
Se, nel vostro programma, avete la linea:

```
PRINT D$; "READ FSEQ"
```

i dati, richiesti dal programma, non saranno più presi da tastiera, ma letti nel file.
I dati vengono letti usando l'istruzione Basic Input.
Per esempio, se, per creare il file sequenziale, avete usato la seguente istruzione:

```
PRINT A$,",",B$,",",C$,",",E$
```

dovrete usare l'istruzione

```
INPUT G$,F$,S$,O$
```

per rileggere il file (stesso numero di parametri nei comandi PRINT e INPUT).
Se l'istruzione Input ha meno parametri dell'istruzione PRINT, si perde un dato nel campo in quanto il prossimo Input utilizza il campo successivo. Ne segue il messaggio d'errore

```
? EXTRA IGNORED
```

Al contrario, se ci sono più parametri nell'istruzione Input di quanti ce ne erano nell'istruzione PRINT, i dati presenti sono assegnati ai primi parametri; alcuni parametri non hanno ricevuto né dati né valori: il DOS leggerà il

campo successivo. Se non esiste, il DOS fa comparire il messaggio END of DATA. Se nel nuovo campo, il numero di dati è inferiore al numero dei parametri ai quali il DOS deve assegnare un valore, esso ripete la stessa operazione col campo successivo. Se il numero di dati corrisponde al numero di parametri ai quali il DOS deve assegnare un valore, non compare alcun tipo di errore. Se invece, ci sono troppi dati nell'ultimo campo letto, il DOS dà l'errore

? EXTRA IGNORED

Avete dunque interesse a mettere lo stesso numero di dati nella corrispondente istruzione PRINT.
Consideriamo i seguenti programmi:

Programma 1: creazione del file sequenziale

```
 10 D$ = CHR$ (4)
 20 PRINT D$;"OPEN FSEQ"
 30 PRINT D$;"DELETE FSEQ"
 40 PRINT D$;"OPEN FSEQ"
 50 PRINT D$;"WRITE FSEQ"
 60 PRINT "GATTO,CANE "
 70 PRINT "FRUTTA,APPLE   ,AQUILA "
 80 PRINT "CAVALLO"
 90 PRINT "D$;"CLOSE FSEQ"
100 END
```

Programma 2: lettura del file

```
 10 D$ = CHR$ (4)
 20 PRINT D$;"OPEN FSEQ"
 30 PRINT D$;"READ FSEQ"
 40 INPUT A$,B$
 50 INPUT C$,D1$,E$
 60 INPUT F$
 70 PRINT D$;"CLOSE"
 80 HOME
 90 PRINT A$,B$,C$,D1$,E$,F$
100 END
```

Programma 3: come programma 2 tranne

```
50 INPUT C$,D1$,E$,F$,G$
```

Programma 4: come programma 2 tranne

```
40 INPUT A$
50 INPUT B$,C$,D1$,E$
```

Fate eseguire il programma 1, poi gli altri tre programmi. Cosa succede? Il programma 2 funziona bene, ma i programmi 3 e 4 terminano con l'errore ? EXTRA IGNORED.
Cercate di capire il motivo rileggendo le suddette spiegazioni.
Il comando READ del DOS è annullato da un errore o da un nuovo comando al DOS: esattamente come il comando WRITE.

1. *Solo l'*APPLESOFT *fornisce la possibilità di usare più di un dato; infatti i Basic dell'*APPLE *non usano la virgola allo stesso modo, cioè come separatore nelle istruzioni* PRINT *e* INPUT. *La virgola non è considerata, dall'istruzione Input dell'Integer Basic, come separatore di stringhe di caratteri.*
2. *Quando si scrive in un file, le virgole usate nelle istruzioni* PRINT *(Integer Basic) non sono prese in considerazione. Le virgole hanno, dunque, lo stesso ruolo dei punti e virgola.*
L'istruzione PRINT "HELLO", "APPLE" *in Integer Basic, memorizza nel file il dato* HELLOAPPLE.

Le due stringhe sono concatenate.

- **Aggiunta di dati in un file sequenziale**

Il comando OPEN del DOS apre il file e posiziona il puntatore sul primo campo. Cosa fare per aggiungere dei dati alla fine di un file?

- Leggete tutti i campi; in questo modo eviterete l'errore END OF DATA e potrete scrivere nuovi dati nel file;
oppure:
- Usate il comando APPEND che sostituisce il comando OPEN (quest'ultimo diventa inutile). Il DOS apre il file e posiziona il puntatore alla fine dell'ultimo campo del file. A questo punto potete scrivere nuovi dati nel file.

Consideriamo i seguenti programmi:

Programma 1:

```
10 D$ = CHR$(4)
20 PRINT D$ ; "OPEN FSEQ"
30 PRINT D$ ; "WRITE FSEQ"
40 PRINT "APPLE   "
50 PRINT D$ ; "CLOSE"
60 END
```

Programma 2:

```
10 D$ = CHR$(4)
20 PRINT D$ ; "APPEND FSEQ"
30 PRINT D$ ; "WRITE FSEQ"
40 PRINT "FDS"
50 PRINT D$ ; "CLOSE"
60 END
```

Programma 3:

```
10 D$ = CHR$(4)
20 PRINT D$ ; "OPEN FSEQ "
30 PRINT D$ ; "READ FSEQ "
40 INPUT A$
50 INPUT B$
60 PRINT D$ ; "CLOSE"
70 PRINT A$, B$
80 END
```

Fate eseguire i programmi 1, 2 e 3 nell'ordine 1, 2, 3. Cosa succede? Il programma 1 crea il file sequenziale e vi scrive il dato "APPLE". Il programma 2 apre il file, si posiziona a fine file (APPEND) e vi scrive il dato "FDS". Il file contiene i dati APPLE e FDS; questo viene verificato dal programma 3. Provate a capirne il motivo rileggendo le suddette spiegazioni.

1. *Se il comando* APPEND *non trova il file, il* DOS *provoca l'errore* FILE NOT FOUND *e ferma l'esecuzione del programma.*
2. *Con il comando* APPEND *possono venire a crearsi i seguenti problemi:*
- Se il vostro file termina al margine di un settore (128 *bytes*), *questo comando posiziona il puntatore all'inizio del file e, in questo modo, avviene la sovrapposizione. Per evitare questo rischio, è necessario, prima della chiusura del file, scrivere alla sua fine un contrassegno di fine file.*
Il DOS *non aggiunge contrassegni di fine file al momento della chiusura perché tutti i settori che attribuisce al file vengono riempiti all'inizio di tali contrassegni. Quando scrivete i* 128 *bytes, tutti i contrassegni vengono annullati.*
- Se il vostro file supera 128 *settori, il comando* APPEND *posiziona il puntatore sul primo settore che segue il contrassegno di fine file che non viene dunque cancellato. Alla prossima occorrenza di* APPEND, *i nuovi dati introdotti saranno persi.*

• Posizionamento del prossimo campo da leggere o scrivere

Il comando POSITION del DOS consente di spostare il puntatore, nel file, di un certo numero di campi. Se il campo corrente è il numero 1 e volete leggere, per esempio, il campo numero 5, dovete inserire l'istruzione:

PRINT D$; "POSITION FSEQ, R4"

Il formato esatto del comando POSITION è il seguente:

POSITION nome file, Rp

dove p è la posizione relativa al campo, sulla quale posizionarsi in relazione al campo corrente (p-esimo campo dopo il corrente). Se p = 0, il puntatore non viene modificato. Se p = 1, il puntatore si posiziona sul campo successivo; ecc.
Se il campo sul quale ci si deve posizionare non esiste (fine file), il DOS provoca l'errore END OF DATA e ferma l'esecuzione del programma. Il comando POSITION analizza il file alla ricerca dei campi. Se viene trovato un carattere vietato (per esempio un blank, codice ASCII 0), compare il suddetto messaggio d'errore.
Consideriamo il seguente esempio (che va assieme ai programmi 1, 2, 3 del paragrafo precedente).

Programma 4:

```
10 D$ = CHR$ (4)
20 PRINT D$;"OPEN FSEQ"
30 PRINT D$;"POSITION FSEQ,R1"
40 PRINT D$;"READ FSEQ"
50 INPUT A$
60 PRINT D$;"CLOSE"
70 HOME
80 PRINT A$
90 END
```

Dopo la creazione del file FSEQ da parte dei programmi 1 e 2, lanciate l'esecuzione del programma 4. Vedrete comparire FDS sullo schermo.
Il comando usa degli spostamenti relativi in relazione al campo. Tuttavia, se il comando POSITION *viene usato prima di qualunque operazione di lettura o scrittura, lo spostamento diventa assoluto (in relazione all'inizio del file).*

• Accesso a un file a partire da un byte specificato

I comandi del DOS: WRITE e READ possono essere usati con un parametro che specifica il byte del file sul quale deve essere posizionato il puntatore. In questo modo, la lettura (o la scrittura) dei dati comincia da questo byte.
L'uso di questo comando vi obbliga a conoscere la struttura esatta dei dati

nel file (returns; e altri caratteri). La numerazione dei bytes nel file inizia da 0. L'opzione B30 porta il posizionamento sul 31-esimo byte del file;

WRITE FSEQ, B30
READ

Questa possibilità può essere utilizzata col comando POSITION. Il byte specificato è, così, situato in relazione all'inizio del campo.
Questa possibilità è da evitare sempre, tranne nel caso in cui i vostri dati seguano un formato fisso oppure abbiate già una certa esperienza.

Esempio: siccome l'uso dei file sequenziali è limitato, vi consigliamo di passare direttamente ai files ad accesso diretto. Per questo motivo non diamo esempi esplicativi sui files sequenziali. Solo l'esempio del paragrafo 1.7.5, che tratta il comando EXEC, userà i files ad accesso sequenziale.

1.6.4 I files ad accesso diretto

• Relazione tra settori fisici e records

I files ad accesso diretto sono files di dati di tipo testo, che si differenziano dai files sequenziali per il fatto che i campi sono di dimensione fissa (records). Se scrivete un record più piccolo della dimensione fissa, esso viene completato con dei blanks (codice ASCII 0).
La struttura delle informazioni sul mini floppy è la seguente:

13 bytes 13 bytes

Sta a voi decidere quale sarà la dimensione dei records. Il DOS assicura la gestione fisica dei settori del mini floppy. Quando la dimensione di un file necessita un nuovo settore fisico, il DOS inizializza il settore con dei blanks. I vostri dati si sovrapporranno ad alcuni di questi blanks.
Il principale vantaggio dei files ad accesso diretto consiste nel fatto che si può accedere ad un record senza aver letto o scritto i records che lo precedono. Siccome la dimensione dei records è fissa, il DOS può calcolare la posizione del record sul mini floppy e accedervi.

• Apertura di un file ad accesso diretto

L'apertura di un file ad accesso diretto si effettua, come quella dei files sequenziali, mediante l'uso del comando OPEN aggiungendo però il parametro L che indica la lunghezza di ogni record.
Se usate il comando:

OPEN FDIRECT, L10

date l'ordine al DOS di aprire il file FDIRECT con una lunghezza di record di 10 bytes oppure, se non esiste, di crearlo.
Il formato del comando OPEN è il seguente:

OPEN file, L1, Ss, Vv, Dd

Il parametro 1 deve essere compreso tra 1 e 32767.
Se il file aperto è stato creato con una lunghezza diversa da quella specificata nel comando OPEN, *quest'ultima non viene presa in considerazione dal* DOS. *Non c'è modo di conoscere la lunghezza usata al momento della creazione del file. Dovete conservare nella vostra documentazione la lunghezza dei records del file. Potete ad esempio includere questo dato nel nome del file (per esempio* FDIRECT:L10*).*
Un'altra soluzione: conservate i dati di gestione del file nel primo record (numero 0*). Considereremo questa seconda soluzione come soluzione standard.*

• **Chiusura di un file ad accesso diretto**

La chiusura di un file si effettua come quella di un file sequenziale. Il comando CLOSE viene utilizzato con gli stessi parametri secondo il seguente formato:

CLOSE file, Dd, Ss, Vv

• **Scrittura in un file ad accesso diretto**

La scrittura dei dati in un file ad accesso diretto si esegue usando l'istruzione PRINT. Come per un file sequenziale, è necessario redirigere i dati verso il DOS, usando il comando WRITE del DOS.
Per scrivere in un file ad accesso diretto, il comando WRITE del DOS ha un parametro supplementare: il numero dei record che volete modificare. Un file ad accesso diretto può effettivamente essere usato senza leggere tutti i records del DOS. Per questo motivo dovete specificare il numero del record sul quale si deve posizionare il puntatore record.
I numeri dei records partono da zero per il primo record del file.
Se usate l'istruzione:

PRINT D$; "WRITE FDIRECT, R4"

il DOS posizionerà il puntatore sul quinto record del file e le successive istruzioni PRINT scriveranno i dati nel quinto record.
Se la lunghezza dei dati scritti in un record è superiore alla lunghezza del record specificata nel comando OPEN, avrete una sovrapposizione dei dati dei records seguenti. Avrete dunque interesse ad evitare questo genere di problema.
Se omettete il numero del record nel comando WRITE, il puntatore non viene modificato e rimane sul record corrente.
Il formato esatto del comando WRITE è il seguente:

WRITE file Rr, Ss, Dd, Vv

Un record di un file ad accesso diretto può contenere uno o più campi (dati) in APPLESOFT. È la stessa cosa utilizzare le istruzioni:

PRINT "BUONGIORNO, COME STATE";
PRINT "APPLE"

e

PRINT "BUONGIORNO,";"COME STATE","APPLE"

In ambedue i casi è stato creato un record contenente tre dati.
Al contrario, un record di un file ad accesso diretto contiene un solo dato in Integer Basic. Nel suddetto caso verrebbe creato il seguente dato:

BUONGIORNO, COME STATE, APPLE

Dopo aver letto questo paragrafo può venirvi in mente una domanda.
Come si fa ad aggiungere dei dati alla fine di un file ad accesso diretto?
Per i files ad accesso diretto, il comando APPEND non esiste e non è applicabile in quanto, quando incontra in blank (codice ASCII nullo), esso ritiene di essere arrivato alla fine del file. Inoltre il blank viene usato per riempire i buchi tra i dati di due records successivi. Dovete dunque garantire la gestione del numero del prossimo record da leggere. La soluzione migliore consiste nel memorizzare questo numero nel primo record (numero 0) del file e rileggerlo quando volete aggiungere un record alla fine del file.
È molto più facile modificare un campo in un file ad accesso diretto che in un file sequenziale. Infatti, potete posizionarvi all'inizio di ogni record e siccome la lunghezza del record è fissa, farete meno fatica a scrivere sui records seguenti.
Se vi trovate a dover caricare molto spesso dei dati (esempio: archivio di indirizzi) avrete più interesse a utilizzare i files ad accesso diretto.

• Lettura di dati in un file ad accesso diretto

La lettura di dati in un file ad accesso diretto si effettua usando l'istruzione INPUT che legge i dati da un file ad accesso diretto nel momento in cui il comando READ del DOS viene attivato. Il suo formato è il seguente:

READ file, Rr, Ss, Dd, Vv

Come per un file sequenziale, il numero dei dati dell'istruzione INPUT dev'essere lo stesso di quello dell'istruzione PRINT corrispondente.
Inoltre, dovete avere sempre lo stesso numero di dati qualunque sia il record in questione. Per esempio, il DOS 3.3 non accetta di scrivere 0 nel record 0, 0 e 1 nel record 1, ecc.; ma si bloccherà alla lettura anche se le istruzioni PRINT e INPUT concordano.
Le modalità d'uso dell'istruzione GET sono identiche.

• Accesso ad un dato byte

Come per un file sequenziale, è possibile posizionare il puntatore del file su un dato byte del file (cfr. par. 1.6.3). Questo si può applicare anche per un byte di un dato record. Per esempio se usate il seguente comando del DOS:

READ FDIRECT, R5, B4

il DOS posizionerà il puntatore sul byte numero 4 (quinto byte del record) del record numero 5 (sesto record del file).

Come per i files sequenziali, sconsigliamo l'uso di questa possibilità ai principianti in quanto occorre conoscere la struttura del file o del campo byte per byte. Tuttavia, è utile se il formato dei campi è fisso.

Esempio: programma per la gestione di un archivio di indirizzi scritto in Basic APPLESOFT, che riprende i sottoprogrammi presentati nel volume 1, capitolo 2 e aggiunge la gestione del file:

Linee 10- 140: inizializzazione
Linee 150- 290: creazione del file se necessario
Linee 310- 400: lettura del numero di records del file
Linee 430- 580: menù
Linee 590- 640: inizio dei sottoprogrammi
Linee 650-1460: ricerca nel file:
- definizione del tipo di ricerca
- percorso del file e test di uguaglianza
- chiamata del sottoprogramma di modifica/soppressione di un record

Linee 1470-1940: aggiunta di record:
- acquisizione dei dati
- scrittura a fine file

Linee 1960-2170: visualizzazione del file
Linee 2180-2230: fine del programma
Linee 2260-2300: inizio dei sottoprogrammi del secondo livello
Linee 2300-2770: modifica/soppressione di un record:
- linee 2300-2430 menù
- linee 2440-2690 modifica
- linee 2700-2770 soppressione

Linee 2850-2920: lettura del file
Linee 2930-2995: scrittura nel file:
- acquisizione del cognome
- acquisizione del nome
- acquisizione del tipo di via
- acquisizione del nome della via
- acquisizione del numero
- acquisizione della città
- acquisizione del codice postale
- acquisizione del prefisso e del numero di telefono

Linee 3860-3950: visualizzazione del messaggio d'errore "NON PIÙ DI N CARATTERI"
Linee 3960-4000: visualizzazione dei dati presi
Linee 4000-4060: inquadratura di un dato tra N spazi

```
10  REM   AGENDA INDIRIZZI
20  REM
30  REM  COPYRIGHT DE MERLY - LAMOITIER
40  REM
50  REM  10/12/82
60  REM
70  REM   CREAZIONE DEL FILE  ?
80  REM
90  HOME : INVERSE
100  PRINT "      AGENDA INDIRIZZI   ": PRINT
110  NORMAL
111  DIM RECH$(6,2),RECH(6,2)
112 C$ = "                    "
114 D$ = ""
115 OPEN$ = D$ + "OPEN FICHAD,L128":CLOSE$ = D$ + "CLOSE FICHAD"
116 RD$ = D$ + "READ FICHAD,R":WR$ = D$ + "WRITE FICHAD,R":DL$ =
                                          D$ + "DELETE FICHAD"
120  INPUT "VUOI CREARE IL FILE (Y/N) ";A$
130  IF A$ <  > "N" AND A$ <  > "Y" THEN 120
140  IF A$ = "N" THEN 320
150  REM
160  REM   CREAZIONE DEL FILE
170  REM
180  PRINT "STO CREANDO  IL FILE PRECEDENTE ": PRINT "E' CANCELLATO"
190  INPUT "VUOI CONTINUARE LO STESSO (Y/N) ";A$
200  PRINT CLOSE$
210  IF A$ = "N" THEN 320
220  PRINT OPEN$: PRINT DL$: PRINT OPEN$
230 CODE = 0
240 NB = 1
270  PRINT WR$ + "0"
280  PRINT  STR$ (NB)
290  PRINT D$ + "CLOSE"
310  REM
320  REM   LETTURA DEL NOME DI REGISTRAZIONE
330  REM   NEL FILE
340  REM
350  PRINT OPEN$
360  PRINT RD$ + "0"
370  INPUT NB$
380  PRINT D$
390 NB =  VAL (NB$)
400 CODE$ = "0"
430  REM
440  REM   VISUALIZZAZIONE DEL MENU
450  REM  *******************
460  REM
470  HOME : INVERSE
480  PRINT "    *** AGENDA INDIRIZZI  ***": PRINT : PRINT : PRINT
490  PRINT "1- VISUALIZZAZIONE CON/SENZA MODIFICHE"
```

```
500  PRINT "2- INSERIMENTO (NUOVO INDIRIZZO ) "
510  PRINT "3- LISTA DEL FILE  "
520  PRINT "4- FINE PROGRAMMA  "
530  PRINT
540  INPUT "SCEGLI UN NUMERO   (1,2,3,4) ";A
550  IF A < 1 OR A > 4 THEN 470
560  NORMAL
570  ON A GOSUB 640,1490,1960,2180
580  GOTO 470
590  REM
600  REM
610  REM   INIZIO SOTTO PROGRAMMI
620  REM  *************************
630  REM
640  REM
650  REM  ************************************
660  REM    RICERCA DENTRO IL FILE    (1)
670  REM  ************************************
680  REM
690  HOME : INVERSE : PRINT "   *** VISUALIZZAZIONE *": NORMAL
700  PRINT : PRINT : PRINT
710  PRINT "DA DOVE INIZIA LA RICERCA        "
720  PRINT
730  PRINT "1-IL COGNOME"
740  PRINT "2-IL NOME"
750  PRINT "3-L'INDIRIZZO"
760  PRINT "4-LA CITTA'"
770  PRINT "5-IL NUMERO DI CODICE POSTALE"
780  PRINT "6-IL NUMERO DI TELEFONO "
790  PRINT : PRINT : INPUT "SCEGLIERE  (INSERIRE ANCHE PIU'
                                                   CIFRE ) ";B
810 RECH$(1,2) = "RICERCA SUL COGNOME"
820 RECH$(2,2) = "RICERCA SUL NOME "
830 RECH$(3,2) = "RICERCA SUL INDIRIZZO"
840 RECH$(4,2) = "RICERCA SULLA CITTA'"
850 RECH$(5,2) = "RICERCA SUL CODICE POSTRALE"
860 RECH$(6,2) = "RICERCA SUL NUMERO DI TELEFONO "
870 RECH(1,2) = 15:RECH(2,2) = 15:RECH(3,2) = 15:RECH(4,2) =
            15:RECH(5,2) =  5:RECH(6,2) = 7
930 B =  LEN (B$)
940  FOR I = 1 TO 6
950 RECH(I,1) = 0
960  FOR J = 1 TO B
970  IF  VAL ( MID$ (B$,J,1)) <  > I THEN 1010
980 RECH(I,1) = 1
990  PRINT RECH$(I,2): INPUT " ";A$
1000 N = RECH(I,2): GOSUB 5000:RECH$(I,1) = A$
1010  NEXT J
1020  NEXT I
```

```
1030  REM
1040  REM   FASE DI RICERCA
1050  REM  ************************
1060  REM
1100  IF NB > 1 THEN 1120
1110  PRINT "FILE VUOTO  ": INPUT "PREMERE RETURN PER CONTINUARE"
      ;A: RETURN
1120  FOR I = 1 TO NB - 1
1130 OK = 1
1140  PRINT RD$ +  STR$ (I)
1150  GOSUB 2910
1160  PRINT D$
1170  REM
1180  REM   TEST DI UGUAGLIANZA
1190  REM  ******************
1200  REM
1210  REM   ARTICOLI -DATI DA RICERCARE
1220  REM  *****************************
1230  REM
1235 OK = 1
1240  IF RECH(1,1) = 0 THEN 1260
1250 OK = OK AND (COGN$ = RECH$(1,1))
1260  IF RECH(2,1) = 0 THEN 1280
1270 OK = OK AND (NOM$ = RECH$(2,1))
1280  IF RECH(3,1) = 0 THEN 1310
1290 A$ = NUMERO$ + " " + TIPO$ + " " + NVIA$
1300 OK = OK AND A$ = RECH$(3,1))
1310  IF RECH(4,1) = 0 THEN 1330
1320 OK = OK AND (CITTA$ = RECH$(4,1))
1330  IF RECH(5,1) = 0 THEN 1350
1340 OK = OK AND (CDPST$ = RECH$(5,1))
1350  IF RECH(6,1) = 0 THEN 1370
1360 OK = OK AND (TEL$ = RECH$(6,1))
1370  IF OK = 0 THEN 1440
1375  PRINT "1375"
1380  HOME : INVERSE : PRINT "DATI POSSIBILI ": NORMAL
1390  PRINT : PRINT : PRINT : GOSUB 4000: PRINT
1400  PRINT "VUOI MODIFICARE O CANCELLARE "
1410  INPUT "QUESTA REGISTRAZIONE (Y/N)";A$
1420  IF A$ <  > "N" AND A$ <  > "Y" THEN 1400
1430  IF A$ = "Y" THEN  GOSUB 2310
1440  NEXT I
1450  RETURN
1470  REM
1480  REM  *********************************
1490  REM       AGGIUNTA DATI  (2)
1500  REM  *********************************
1510  REM
1520  INVERSE
```

```
1530  HOME : PRINT "NUOVO INDIRIZZO ": PRINT : PRINT
1540  NORMAL
1550  PRINT "SE UN DATO NON PUO' ESSERE INSERITO"
1560  PRINT "PREMERE SOLO  RETURN  "
1570  PRINT
1580 CODE$ = "1"
1590  GOSUB 3000: REM    SCELTA DEL COGNOME
1595 B$ = NOM$
1600  GOSUB 3100: REM   SCELTE DEL NOME
1610  GOSUB 3200: REM   SCELTA DEL TIPO DI VIA
1620  GOSUB 3300: REM   SCELTA DEL NOME DELLA VIA
1630  GOSUB 3400: REM   SCELTA DEL NUMERO
1640  GOSUB 3500: REM   SCELTA DELLA CITTA'
1650  GOSUB 3600: REM   SCELTA DEL CODICE POSTALE
1660  GOSUB 3700: REM   SCELTA DEL TELEFONO
1670  HOME
1680  INVERSE
1690  PRINT "INFORMAZIONI SCELTE "
1700  NORMAL : PRINT : PRINT
1710  GOSUB 4000: REM   AGGIUNTA NEL BUFFER
1720  PRINT : INVERSE
1730  INPUT "CONVALIDI QUESTE INFORMAZIONI (Y/N)";A$
1740  IF A$ <  > "N" AND A$ <  > "Y" THEN 1730
1745  IF A$ = "N" THEN 1490
1750  REM
1760  REM   AGGIUNTA DEL RECORD NEL FILE
1770  REM
1780  PRINT WR$;NB
1790  GOSUB 2990
1800 NB = NB + 1
1810  PRINT WR$ + "0": PRINT  STR$ (NB)
1820  PRINT D$
1940  INPUT "PREMERE RETURN PER CONTINUARE";A$
1950  RETURN
1960  REM
1970  REM  ******************
1980  REM   LISTING DEL FILE
1990  REM  ******************
2000  REM
2010  IF NB = 1 THEN 2150
2040  FOR I = 1 TO NB - 1
2050  PRINT RD$;I
2060  GOSUB 2910
2061  PRINT D$
2070  IF CODE$ <  > "1" THEN 2140
2080  HOME : INVERSE
2090  PRINT "RECORD NUMERO  ";I
2100  NORMAL
2110  PRINT
```

```
2120  GOSUB 4000
2130  INPUT "PREMERE RETURN PER CONTINUARE";A$
2140  NEXT
2150  HOME : INVERSE : PRINT "FINE DEL FILE ": NORMAL
2160  INPUT "PREMERE RETURN PER CONTINUARE";A$
2170  RETURN
2180  REM
2190  REM  ***********************************
2200  REM       FINE DEL PROGRAMMA (4)
2210  REM  ***********************************
2220  REM
2225  PRINT WR$ + "0": PRINT  STR$ (NB)
2230  PRINT D$ + "CLOSE"
2240  NORMAL
2250  END
2260  REM
2270  REM   INIZIO SOTTOPROGRAMMI
2280  REM   DEL SECONDO LIVELLO
2290  REM  *************************
2300  REM
2310  REM
2320  REM   MODIFICA / CANCELLAZIONE
2330  REM  ************************
2340  REM
2350  REM   DI UNA REGISTRAZIONE
2360  REM  *******************
2370  REM
2380  PRINT : PRINT "VUOI"
2390  PRINT "MODIFICARE IL RECORD (1)"
2400  PRINT "CANCELLARE IL RECORD (2)"
2410  INPUT "SCEGLI ";A
2420  IF A < 1 OR A > 2 THEN 2380
2430  ON A GOTO 2440,2700
2440  REM
2450  REM   MODIFICA DI UN RECORD
2460  REM  *******************
2470  REM
2480  PRINT : PRINT "CHE COSA VUOI MODIFICARE"
2490  PRINT : PRINT "1-IL COGNOME": PRINT "2-IL NOME  ": PRINT
      "3-L'INDIRIZZO": PRINT "4-LA CITTA'": PRINT "5-IL CODICE POSTALE":
      PRINT "6-IL NUMERO DI TELEFONO"
2500  PRINT : INPUT "SCEGLI UN NUMERO (1,2,3,4,5,6) ";A
2510  IF A <  > 3 THEN 2560
2520  GOSUB 3200: REM   TIPO DI VIA
2530  GOSUB 3300: REM   NOME DELLA VIA
2540  GOSUB 3400: REM   NUMERO
2550  GOTO 2570
2560  ON A GOSUB 3000,3100,,3470,3570,3670
2570  HOME : INVERSE : PRINT "INFORMAZIONI SCELTE ": NORMAL
```

```
2580  GOSUB 4060
2590  PRINT : PRINT "CHE SCEGLI"
2600  PRINT "1-MODIFICARE UN ALTRO DATO DEL RECORD"
2610  PRINT "2-INSERIRE LE MODIFICHE"
2620  PRINT "3-NON INSERIRE LE MODIFICHE"
2630  INPUT "SCEGLI UN NUMERO (1,2,3)";A
2640  ON A GOTO 2440,2650,2690
2650  REM
2660  REM   REGISTRAZIONE DELLE MODIFICHE
2670  REM
2680  PRINT WR$;I: GOSUB 2990: PRINT D$
2690  RETURN
2700  REM
2710  REM   CANCELLAZIONE DI UN RECORD
2720  REM  ************************
2730  REM
2740  PRINT WR$;I
2750 CODE$ = "0"
2760  GOSUB 2990
2770  PRINT D$
2850  REM
2860  REM    LETTURA DI UN FILE
2870  REM    ***************
2880  REM
2910  INPUT B$
2911 CODE$ =  MID$ (B$,1,1):COGN$ =  MID$ (B$,2,15):NOM$ =
     MID$ (B$,17,15)
2912 TIPO$ =  MID$ (B$,32,3):NVIA$ =  MID$ (B$,35,15)
     :NUMERO$ =  MID$ (B$,50,3)
2913 CITTA$ =  MID$ (B$,53,15):CDPST$ =  MID$ (B$,68,5)
2914 INDTEL$ =  MID$ (B$,73,2):TEL$ =  MID$ (B$,75,7)
2920  RETURN
2930  REM
2940  REM   SCRITTURA DI UN FILE
2950  REM  ************************
2960  REM
2990  PRINT CODE$;COGN$;NOM$;TIPO$;NVIA$;NUMERO$;CITTA$;CDPST$;
      INDTEL$;TEL$
2995  RETURN
3000  REM
3010  REM   SCELTA DEL COGNOME
3020  REM  *************
3030  REM
3040  INPUT "COGNOME ";A$
3050 N = 15: IF  LEN (A$) <  = N THEN 3080
3060  GOSUB 3920: REM   MESSAGGIO DI ERRORE
3070  GOTO 3040
3080  GOSUB 5000:COGN$ = A$
3090  RETURN
```

```
3100  REM
3110  REM   SCELTA DEL NOME
3120  REM  ****************
3130  REM
3140  INPUT " NOME  ";A$
3150 N = 15: IF  LEN (A$) <  = N THEN 3180
3160  GOSUB 3920: REM   MESSAGGIO DI ERRORE
3170  GOTO 3140
3180  GOSUB 5000:NOM$ = A$
3190  RETURN
3200  REM
3210  REM   SCELTA DEL TIPO DI STRADA
3220  REM  ***********************
3230  REM
3240  INPUT "TIPO DI STRADA (VIA,CORSO...) ";A$
3250 N = 3: IF  LEN (A$) <  = N THEN 3280
3260  GOSUB 3920: REM   MESSAGGIO DI ERRORE
3270  GOTO 3240
3280  GOSUB 5000:TIPO$ = A$
3290  RETURN
3300  REM
3310  REM   SCELTA DEL NOME DELLA VIA
3320  REM  ***********************
3330  REM
3340  INPUT "NOME DELLA VIA ";A$
3350 N = 15: IF  LEN (A$) <  = N THEN 3380
3360  GOSUB 3920: REM   MESSAGGIO DI ERRORE
3370  GOTO 3340
3380  GOSUB 5000:NVIA$ = A$
3390  RETURN
3400  REM
3410  REM   SCELTA DEL NUMERO
3420  REM  ****************
3430  REM
3440  INPUT "NUMERO ";A$
3450 N = 3: GOSUB 5000:NUMERO$ = A$
3460  RETURN
3470  REM
3480  REM   SCELTA DELLA CITTA'
3490  REM  ******************
3500  REM
3510  INPUT "CITTA' ";A$
3520 N = 15: IF  LEN (A$) <  = N THEN 3550
3530  GOSUB 3870: REM   MESSAGGIO DI ERRORE
3540  GOTO 3510
3550  GOSUB 5000:CITTA$ = A$
3560  RETURN
3570  REM
3580  REM   SCELTA DEL CODICE POSTALE
```

```
3590  REM  ********************
3600  REM
3610  INPUT "CODICE POSTALE";A$
3620 N = 5: IF  LEN (A$) <  = N THEN 3650
3630  PRINT "NON PIU' DI CINQUE CIFRE "
3640  GOTO 3610
3650  GOSUB 5000:CDPST$ = A$
3660  RETURN
3670  REM
3680  REM    SCELTA DEL PREFISSO
3690  REM  *********************************
3700  REM
3710  INPUT "PREFISSO TELEFONICO ";A$
3720 N = 2: IF  LEN (A$) <  = N THEN 3750
3730  PRINT "NON PIU' DI DUE CIFRE"
3740  GOTO 3710
3750 N = 2: GOSUB 5000:INDTEL$ = A$
3760  REM
3770  REM   SCELTA DEL NUMERO DI TELEFONO
3780  REM  ****************************
3790  REM
3800  INPUT "NUMERO DI TELEFONO ";A$
3810 N = 7: IF  LEN (A$) <  = N THEN 3840
3820  PRINT "NON PIU' DI SETTE CIFRE"
3830  GOTO 3800
3840  GOSUB 5000:TEL$ = A$
3850  RETURN
3860  REM
3870  REM
3880  REM   VISUALIZZAZIONE DEI MESSAGGI D'ERRORE
3890  REM   STRINGA TROPPO LUNGA
3900  REM  *************************
3910  REM
3920  PRINT : INVERSE
3930  PRINT "NON PIU' DI ";N;" CARATTERI,PREGO"
3940  NORMAL
3950  RETURN
3960  REM
3970  REM   VISUALIZZAZIONE DEI DATI SCELTI
3980  REM  ****************************
3990  REM
4000  PR#1
4010  PRINT "COGNOME"; TAB( 20);COGN$
4020  PRINT "NOME"; TAB( 20);NOM$
4030  PRINT "INDIRIZZO"; TAB( 20);NUMERO$;" ";TIPO$;" ";NVIA$
4040  PRINT "CITTA'"; TAB( 20);CDPST$;" ";CITTA$
4050  PRINT "TELEFONO "; TAB( 20);"(";INDTEL$;")";TEL$
4060  RETURN
5000  REM
```

```
5010  REM   QUADRO DEI DATI SCELTI
5020  REM  ******************************
5030  REM
5040  REM   IN UNA CATENA DI N CARATTERI
5050  REM  ******************************
5055 N = N -  LEN (A$)
5056  IF N <  = 0 THEN  RETURN
5060 A$ = A$ +  LEFT$ (C$,N)
5090  RETURN
```

1.7 I COMANDI CATALOGATI SULL'APPLE

1.7.1 Presentazione

Potete creare dei comandi catalogati sul vostro APPLE; per farlo, dovete creare un file sequenziale (cf. par. 1.6.3) contenente i comandi da eseguire. Supponiamo, ad esempio, che il file FCOM contenga i campi seguenti:

```
 5 HOME
10 PRINT "COUCOU"
20 END
   LIST
   RUN
   HOME
   CATALOG
```

Come lanciare il comando catalogato FCOM? Sarà sufficiente digitare il seguente comando:

```
EXEC FCOM
```

Cosa succede?
Le linee Basic 5, 10, 20 vengono caricate in memoria. Il programma viene dunque visualizzato ed eseguito. Lo schermo si sbianca e compare il catalogo del mini floppy.

1.7.2 Esempio: cambio del tipo di un file Basic

Il comando EXEC carica in memoria eventuali linee Basic numerate che si trovano nel file sequenziale del comando catalogato. Questo è un modo per caricare un programma Basic da un file di tipo testo.
Consideriamo il seguente programma:

```
10 D$=CHR$(4)
20 PRINT D$;"OPEN LISTING"
30 PRINT D$;"WRITE LISTING"
35 POKE 33,33:REM LINEA DI 33 CARATTERI
```

```
40 LIST
50 PRINT D$;"CLOSE"
60 END
```

Questo programma crea il file LISTING che contiene il listing del programma del quale si vuole caricare il tipo. Cambiamo Basic e digitiamo il comando EXEC LISTING. Il file viene ricaricato in memoria. Non rimane che memorizzare il programma in memoria in un file dell'altro tipo.
Questo esempio può servire ad aggiungere ai vostri programmi dei sottoprogrammi standard, a modificare i numeri delle linee di una parte di un programma oppure per recuperare una parte di programma che ha avuto dei problemi (mini floppy danneggiato...).

1.7.3 Esempio: trasformazione di un programma in linguaggio macchina in un programma Basic

Per eseguire la trasformazione, è sufficiente leggere il contenuto di un indirizzo e scrivere linee del tipo POKE a, b in un file sequenziale. In seguito, queste linee saranno caricate in memoria dal comando EXEC.

1.7.4 Formato del comando EXEC

Il formato del comando EXEC è il seguente:

EXEC nome file

Il file è un file sequenziale contenente dei comandi Basic o delle linee di programmi.
I campi vengono letti e, se si tratta di comandi Basic, vengono eseguiti; se si tratta di linee di programma, vengono caricati in memoria. Il comando catalogato termina quando non ci sono più campi da trattare.
Quando un campo non può essere interpretato, il DOS fa comparire il messaggio SYNTAX ERROR e passa al trattamento del campo seguente.
Se volete fermare l'esecuzione del comando catalogato attivato, dovrete battere il tasto RESET in quanto il CTRL-C non ha effetto.
Potete mettere il comando RUN *in un comando catalogato. Dovrete però essere prudenti in quanto ogni istruzione* INPUT *(o* GET*) si riferirà al file dei comandi catalogati (come se il comando* READ *fosse attivato). Quando viene incontrata l'istruzione* INPUT, *il prossimo comando viene letto ed eseguito. Il risultato si rivelerà immediatamente molto complicato.*
Se volete fermare l'esecuzione di un programma Basic, potete usare il CTRL-C, *ma verrà bloccata anche l'esecuzione del comando catalogato.*
Due comandi catalogati non possono essere attivati contemporaneamente. Se durante l'esecuzione di un comando catalogato il DOS *trova il comando* EXEC *file* 2, *viene fermato il primo comando e attivato il secondo.*
L'esecuzione di un comando catalogato può iniziare dal p-esimo comando. Digitate:

EXEC file, Rp

I comandi sono numerati a partire da 0. Usare R0 è inutile in quanto lo 0-esimo comando è situato all'inizio del file. Se usate R1, saltate il primo comando (n = 0), e l'esecuzione del comando catalogato inizierà dal secondo comando (n = 1).

La posizione p *specifica è sempre assoluta (cioè è in relazione all'inizio del file).*

1.7.5 Esempio

Riprendiamo qui sotto l'esempio, presentato nel volume 1, paragrafo 3.3, del grafico che rappresenta le curve Y = f(X) in coordinate cartesiane. La modifica consiste nel fatto che la linea 1280 viene caricata automaticamente in memoria e che l'equazione digitata può essere richiamata dal comando EXEC EQUATION quando il programma è in memoria.
La struttura del programma è la seguente:

Linee 10- 150: creazione della linea 1280 e del comando catalogato EQUATION che viene eseguito immediatamente
Linee 184- 250: inizializzazione dei parametri della curva
Linee 255- 440: calcolo della scala e degli assi
Linee 441- 480: visualizzazione dell'equazione
Linee 481- 510: tracciato degli assi
Linee 511- 550: visualizzazione del punto di partenza
Linee 551- 600: tracciato della curva
Linee 600- 690: tracciato della scala dell'asse delle X
Linee 691- 760: tracciato della scala dell'asse delle Y
Linee 761- 880: sottoprogramma per la determinazione della scala
Linee 1009-2660: trattamento degli errori

```
10  REM  PROGRAMMA DI TRACCIAMENTO CURVE
20  REM
30  REM  COPYRIGHT DE MERLY
40  REM
50  TEXT : HOME
60  PRINT "QUESTO PROGRAMMA TRACCIA DELLE CURVE"
70  PRINT "UTILIZZABILE CON LE FUNZIONI"
80  PRINT "MATEMATICHE DELL'APPLESOFT."
90  PRINT "QUINDI SCRIVETE LA VOSTRA ESPRESSIONE"
100  PRINT "CON IL SEGUENTE FORMATO:"
110  PRINT " 1010 Y=F(X)"
120  PRINT
130  PRINT " ESEMPIO: 1010 Y=EXP(X)"
140  PRINT : PRINT "SCRIVETE  RUN 170"
150  END
170  PRINT "SCRIVETE L'EQUAZIONE SECONDO LO STESSO FORMATO "
175  INPUT E$
```

```
180  DEF  FN I(X) =  INT (X + (1 -  SGN (X)) / 2)
184  REM
185  REM  PARAMETRI DELLA CURVA
186  REM
190  HOME
200  PRINT E$
210  PRINT : PRINT ""
220  INPUT "X INIZIALE -->";XD
230  PRINT
240  INPUT "X FINALE   -->";XF
250  ONERR  GOTO 2500
255  REM
256  REM  CALCOLO DELLA SCALA E DEGLI ASSI
257  REM
260  IF XF < XD THEN T = XF:XF = XD:XD = T
270  IF XD <  = 0 AND XF > 0 THEN XL = XF - XD:XC =  - XD
280  IF XD < 0 AND XF <  = 0 THEN XL =  - XD:XC =  - XD
290  IF XD > 0 AND XF > 0 THEN XL = XF:XC = 0
300 XI = XL / 279
310 XZ = XC / XI
320  PRINT : PRINT "OPERAZIONE IN CORSO"
330 YD = 0:YF = 0
340  FOR X = XD TO XF STEP XI
350  GOSUB 1010
360  IF Y > YF THEN YF = Y
370  IF Y < YD THEN YD = Y
380  NEXT X
390  IF YD <  = 0 AND YF > 0 THEN YL = YF - YD:YC =  - YD
400  IF YD < 0 AND YF <  = 0 THEN YL =  - YD:YC =  - YD
410  IF YD > 0 AND YF > 0 THEN YL = YF:YC = 0
420 YI = YL / 159
430 YZ = YC / YI
440 YZ = 159 -  INT (YZ)
441  REM
442  REM  VISUALIZZAZIONE DELL'EQUAZIONE
443  REM
450  HGR
460  VTAB 22
470  IF  LEN (E$) < 40 THEN  HTAB (40 -  LEN (E$)) / 2
480  PRINT E$
481  REM
482  REM  TRACCIAMENTO DEGLI ASSI
483  REM
490  HCOLOR= 3
500  HPLOT 0,YZ TO 279,YZ
510  HPLOT XZ,0 TO XZ,159
511  REM
512  REM  PUNTO DI PARTENZA
513  REM
```

```
520 X = XD
530  GOSUB 1010
540 Y = 159 - (Y + YC) / YI
550  HPLOT (X + XC) / XI,Y
551  REM
552  REM  TTRACCIAMENTO DELLA CURVA
553  REM
560  FOR X = XD + XI TO XF STEP XI
570  GOSUB 1010
580 Y = 159 - (YC + Y) / YI
590  HPLOT  TO (X + XC) / XI,Y
600  NEXT X
601  REM
602  REM  TRACCIAMENTO SCALA DELL'ASSE X
603  REM
610 R = XL
620  GOSUB 770
630  HCOLOR= 0
640  IF XD < 0 AND XF < 0 THEN XF = 0
650  IF XD > 0 AND XF > 0 THEN XD = 0
660  FOR X =  FN I(XD / DI) * DI TO  FN I(XF / DI) * DI STEP DI
670  HPLOT XZ + X / XI,YZ
680  NEXT X
690  VTAB 24: PRINT DI;"/ DIVISIONE PER X";
691  REM
692  REM  TRACCIAMENTO DELL'ASSE DELLE Y
693  REM
700 R = YL
710  GOSUB 770
720  FOR Y =  FN I(YD / DI) * DI TO  FN I(YF / DI) * DI STEP DI
730  HPLOT XZ,YZ - Y / YI
740  NEXT Y
750  PRINT  SPC( 2);DI;"/DIVISIONE PER Y";
760  GOTO 2650
761  REM
762  REM  DETERMINAZIONE DELLA SCALA
763  REM
770 K = 0
780  IF R < 1 THEN 840
790  IF R < 10 THEN 870
800 K = K + 1
810 R = R / 10
820  GOTO 790
830  IF R > 1 THEN 870
840 K = K - 1
850 R = R * 10
860  GOTO 830
870 DI =  INT (R) * 10 ^ (K - 1)
880  RETURN
```

```
1009  REM ***********************
1010 Y = 30
1011  REM ***********************
1020  RETURN
2400  REM
2410  REM  GESTIONE DEGLI ERRORI
2420  REM
2500  POKE 216,0: REM  ANNULLA ONERR
2510 EA =  PEEK (218) +  PEEK (219) * 256
2520 E2$ = "ERRORE"
2530 N$ = "":E1$ = ""
2540  IF EA <  > 1010 THEN N$ = "NON E'  ":E1$ = " ALLA LINEA " +
                                                           STR$ (EA)
2550  HOME : VTAB 21
2560 ER =  PEEK (222)
2570  IF E$ = "" THEN  PRINT "ERRORE NIENTE EQUAZIONE.": END
2580  PRINT "ERRORE DI SINTASSI ";N$;"NELL'EQUAZIONE."
2590  PRINT E$
2600  IF EA <  > 1010 THEN 2620
2610  IF ER = 16 OR ER = 163 OR ER = 224 THEN E2$ = "EQUAZIONE
                                                    SBAGLIATA"
2620  IF ER = 53 OR ER = 69 OR ER = 133 THEN E2$ = "DATI SBAGLIATI"
2630  IF ER = 255 THEN E2$ = "CONTROL-C"
2640  PRINT E2$ + E1$
2650  VTAB (15): PRINT  CHR$ (7): GET N$: TEXT
2660  END
```

1.8 PROGRAMMI DI UTILITÀ DEL DOS 3.3

1.8.1 Copia di un mini floppy

È sempre preferibile conservare una copia dei vostri mini floppy in quanto può sempre verificarsi un incidente.
Un programma vi consente di ricopiare interamente un mini floppy: i programmi COPY (Integer Basic) o COPYA (APPLESOFT) che si trovano sul mini floppy SYSTEM MASTER.
Quando lanciate la copia di un mini floppy, il programma vi chiede la situazione dei mini floppy (slot, drive) secondo il seguente formato:

```
APPLE DISKETTE DUPLICATION PROGRAM

ORIGINAL SLOT:DEFAULT=6
DRIVE:DEFAULT=1
DUPLICATE SLOT:DEFAULT=6
DRIVE:DEFAULT=2

--PRESS "RETURN" KEY TO BEGIN COPY
```

Ad ognuna delle domande riguardanti gli slot e i drive, dovete digitare:

Return se l'opzione per default vi conviene, oppure il numero voluto e poi Return.
Quando il programma vi chiede di battere il Return per cominciare la copiatura inserite il mini floppy da copiare nel primo drive specificato e un mini floppy vergine, o da riscrivere, nel secondo drive.
La copiatura inizia. Il mini floppy viene inizializzato e tutti i files vengono ricopiati. Quando la trascrizione è terminata, il programma vi chiede se volete fare un'altra copia.
Nel caso abbiate un sistema con un solo drive, il programma vi chiederà di mettere successivamente il mini floppy da copiare e il mini floppy sul quale si fa la copia.
Per evitare una falsa manipolazione, vi raccomandiamo di proteggere in scrittura il mini floppy da copiare (cfr. par. 1.2).

1.8.2 Copia di files. Il programma FID

• Presentazione

Il programma FID vi consente di copiare dei files su più mini floppy, di ottenere il catalogo, di distruggere un file, di bloccare un file e di conoscere qual è lo spazio disponibile su un mini floppy.
Per usare questo programma, il mini floppy SYSTEM MASTER dev'essere nel drive attivo. Dovete dunque digitare il comando BRUN FID.
A questo punto compare sullo schermo una lista di comandi preceduti da un numero racchiuso tra < >. Per eseguire uno dei comandi, digitate il numero del comando e Return.

• Designazione dei files

Alcuni comandi del FID vi richiederanno un nome di file, che avrà lo stesso formato che per i comandi del DOS, ma sarà possibile sostituirne una parte col simbolo =. Saranno così considerati tutti i files il cui nome è compreso nel formato specificato.
Per esempio, se digitate:

COLOR=

saranno presi in considerazione tutti i files il cui nome comincia per COLOR.
Se viene usato il simbolo =, compare la seguente domanda:

DO YOU WANT PROMPTING?

Se rispondete Y (Yes, sì), il programma vi chiederà, per ogni file corrispondente al nome dato, se volete eseguire o meno il comando. Dovrete rispondere Y (sì) o N (no) oppure Q (abbandonare). Se digitate Y, il file verrà trattato dal comando. Se digitate N, il file non verrà trattato. Se digitate Q, il comando viene abbandonato.
Se rispondete N (no) alla domanda DO YOU WANT PROMPTING? tutti i files, il cui nome corrisponde, saranno trattati dal comando attivo.

• Copia di un file

Se avete risposto 1 al menù principale del FID, avete selezionato il comando di copiatura di un file. Il programma vi chiederà le caratteristiche dei drives (stesso principio del par. 1.8.1), e il nome del file (cfr. *Designazione dei files*). Il comando vi chiederà poi di inserire i mini floppy e battere ESC, se volete fermare la copiatura, battere qualunque tasto tranne RESET, ESC, CTRL, SHIFT, per continuare.
Se i files di cui avete chiesto la copia esistono e c'è spazio sufficiente sul nuovo mini floppy, vedrete apparire sullo schermo il messaggio DONE. Altrimenti, se nessun file sul mini floppy originale corrisponde a quello che avete richiesto, compare il messaggio NO FILES SELECTED.
Se il mini floppy di destinazione contiene un file con lo stesso nome di quello da ricopiare, compare il seguente messaggio:

FILE ALREADY EXISTS
TYPE IN NEW FILE NAME FOR THE COPY OR
<RETURN> TO REPLACE PRESENT FILE OR
<CTRL-C> <RETURN> TO CANCEL COPY

Vi si offrono tre possibilità:

- ridenominare il file;
- battere Return per sostituire il vecchio file con quello nuovo;
- battere CTRL-C Return per non ricopiare il file.

Se nella seconda soluzione il vecchio file è bloccato, compare sullo schermo il seguente messaggio:

FILE LOCKED
DO YOU WISH TO REPLACE IT ANYWAY?

Se digitate N (no), il suddetto messaggio ricompare. Se digitate Y (sì), il vecchio file viene sostituito.
È possibile copiare un file tra due mini floppy anche se il vostro APPLE II ha un solo drive. Dovrete solamente seguire gli ordini comparsi per inserire i mini floppy.

• Catalogo di un mini floppy

Se avete risposto 2 al menù principale, compare sullo schermo il catalogo del mini floppy corrente. Se il mini floppy corrente non è stato definito, il comando, prima di visualizzare il catalogo, vi chiederà lo slot e il numero del drive.

• Spazio disponibile su un mini floppy

Se avete risposto 3 al menù principale, compare sullo schermo il numero di settori utilizzato e di settori disponibili.

• Eliminazione della protezione di un file

Per sbloccare un file, dovete rispondere al menù principale del FID. Il comando vi chiede il nome del file, vi indica di inserire il mini floppy e vi domanda: volete realmente sbloccare il file?
Se il file non è sul mini floppy, compare il messaggio:

NO FILE SELECTED

• Protezione di un file

Il principio è lo stesso del precedente. L'unica differenza consiste nel fatto che dovete rispondere 5 (al posto di 4) al menù principale del FID.

• Distruzione di un file

Questo comando consente di distruggere uno o più files. Per usarlo occorre rispondere 6 al menù principale del FID e dare il nome del file. Se tutto funziona correttamente, compare il messaggio <DONE>. Se non viene trovato nessun file, compare il messaggio NO FILES SELECTED. Se uno dei files è bloccato, compare il messaggio FILE LOCKED e il file non viene distrutto.
Quando viene visualizzato il messaggio PRESS ANY KEY TO CONTINUE, dovete battere un tasto qualunque tranne Reset, CTRL, SHIFT, RETURN.

• Reset delle caratteristiche del mini floppy attivo

Se rispondete 7 al menù principale del FID, vi saranno chiesti nuovamente i numeri di slot e drive dell'unità disco attiva al momento della prossima esecuzione di un comando che ne fa uso.

• Verifica fisica di un file

Il comando 8 del FID è identico al comando VERIFY del DOS. Vi chiede un nome di file e verifica i settori fisici occupati dai files corrispondenti al nome specificato.
Quando un file è stato verificato, il comando visualizza il suo nome e il messaggio DONE. Altrimenti, se il file non può essere verificato, ne risulta l'errore I/O ERROR. Il programma attende che battiate un tasto per ritornare al menù principale del FID.

1.8.3 Creazione di un mini floppy System Master

I mini floppy creati mediante il comando d'inizializzazione INIT del DOS dipendono dalla dimensione del sistema nel quale esso è stato creato. Il programma MASTER CREATE, situato nel mini floppy SYSTEM MASTER, vi permette di creare dei mini floppy "master" adattati alla dimensione della memoria del sistema utilizzato.
Per usare questo programma, digitate il comando:

BRUN MASTER CREATE

Sullo schermo compare prima il messaggio:

DOS 3.3 MASTER-CREATE UTILITY
COPYRIGHT 1980 BY APPLE COMPUTER INC
ALL RIGHT RESERVED
(NOW LOADING DOS IMAGE)

e poi il messaggio

PLEASE INPUT THE GREETING PROGRAM'S
FILE NAME

Digitate ora il nome del programma che verrà eseguito al momento del caricamento del DOS. Questo programma, a differenza del comando INIT, deve già esistere sul mini floppy. Vedrete comparire il messaggio:

REMEMBER THAT MASTER DOES NOT CREATE
THE GREETING PROGRAM, OR PLACE IT IN
THE DISK DIRECTORY
THIS IS THE FILE NAME THAT WILL BE
PLACED WHITHIN THE IMAGE:
HELLO
PLACE THE DISKETTE TO BE MASTERED IN
THE DISK DRIVE
PRESS <RETURN> WHEN READY
NOTE: IF YOU WANT A DIFFERENT FILE NAME,
PRESS <ESC>

Se siete pronti battete Return.
Quando compare il prompt, prima di fare qualunque altro lavoro, ricaricate il DOS. Il vostro mini floppy è diventato un master.

1.8.4 Passaggio di files del vecchio DOS al DOS 3.3: MUFFIN

Il programma MUFFIN, che si trova sul'mini floppy SYSTEM MASTER, permette di trasferire i vostri programmi su un mini floppy DOS 3.3. Per lanciarlo digitate:

BRUN MUFFIN

Vedrete apparire sullo schermo il messaggio

```
****************************************
*  APPLE II DOS 3.2 TO 3.3 CONVERTER   *
*                                      *
*           MUFFIN VERSION D           *
*                                      *
*  COPYRIGHT 1979 APPLE COMPUTER INC.  *
****************************************
CHOOSE ONE OF THE FOLLOWING OPTIONS
        <1>   CONVERT FILES
        <2>   QUIT
WHICH WOULD YOU LIKE?
```

Per iniziare la conversione, battete 1. Compariranno sullo schermo le seguenti domande:

SOURCE SLOT?	slot di partenza
DRIVE?	drive (1, 2) di partenza
DESTINATION SLOT?	slot di destinazione
DRIVE?	drive di destinazione
FILENAME?	nome del file da trasferire

Usando il carattere =, che permette di sostituire uno o più caratteri, è possibile trasferire diversi files contemporaneamente.
Per esempio, se avete i files FICH 11, FICH 12, FICH 21, il nome FICH = 1 consentirà di designare in una volta FICH 11 e FICH 21.

DO YOU WANT PROMPTING?

Il DOS vi domanda se volete verificare il trasferimento (o no). Se battete Y (sì), per ogni possibile file il DOS vi chiederà se il trasferimento è da fare. Se battete N (no), verranno trasferiti tutti i possibili files.
Il DOS in seguito visualizza:

INSERT DISKS THEN PRESS <ESC> TO RETURN
TO MAIN MENU OR ANOTHER KEY TO BEGIN

Inserite i vostri mini floppy e battete RETUN. Se desiderate fermarvi, battete ESC; ritornerete così al menù principale.
Potete utilizzare MUFFIN *con un solo drive, mettendo gli stessi valori per gli slots e di drives di partenza e di destinazione.*
A questo punto, il DOS *specifica i mini floppy da inserire (partenza e arrivo).*

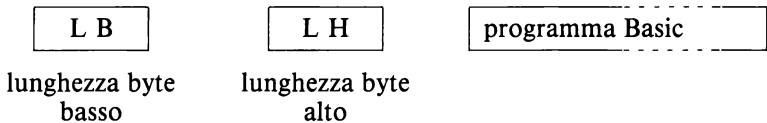
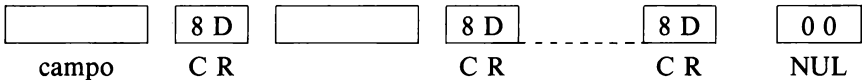
1.9 STRUTTURA INTERNA DEL DOS; STRUTTURA DELLE INFORMAZIONI SU MINI FLOPPY E SOTTOPROGRAMMI UTILIZZABILI

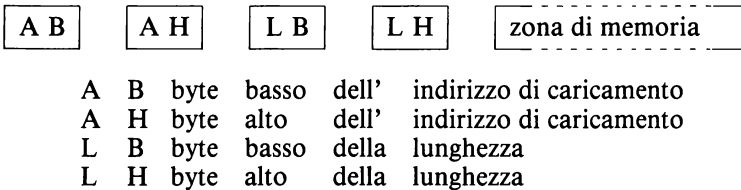
1.9.1 Introduzione

Descriveremo come il DOS memorizza i files sul mini floppy, gestisce il catalogo, ecc.
Su un mini floppy, le informazioni sono registrate in 35 piste numerate da $00 (esterna) a $22 (interna). Queste piste sono suddivise in 16 settori di 256 bytes. Per accedere ad un dato settore di una data pista, il DOS parte facendo girare il mini floppy, e invia il segnale necessario per posizionare la testina di lettura sulla pista desiderata. I diversi settori passano sotto la testina fino al settore voluto.
Per scrivere su un settore, il DOS raccoglie le informazioni in un buffer di 256 bytes, accede al settore sul mini floppy e memorizza i dati.
In generale, il DOS comincia la memorizzazione di un programma o di un file all'inizio di un settore libero. Come avviene la gestione dei settori liberi?

Allo scopo esiste, per ogni pista, una tabella che indica i settori della pista usati (Track bit map). La descriveremo in seguito a questo paragrafo.
Quando un settore è pieno e restano dei dati da memorizzare, il DOS conserva le informazioni riguardanti il settore (numero di pista da 0 a 34, numero di settori nella pista da 0 a 15) in una tabella chiamata "Track Sector List" (elenco dei settori delle piste). Per continuare la memorizzazione del file, usa un altro settore libero. Quando il file è stato interamente registrato, il DOS memorizza la tavola dei settori del file in un nuovo settore di cui conserva la posizione al catalogo del mini floppy.
Il catalogo del disco è memorizzato in un certo numero di settori "concatenati" tra loro. L'indirizzo (pista, settore) del catalogo è memorizzato nella tavola dei contenuti del disco situata nel settore 0 della pista numero 17 ($11). Questa tavola (VTOC) è descritta in dettaglio in seguito a questo paragrafo.

1.9.2 Formato dei files

* BASIC (Integer o APPLESOFT)

Ogni campo contiene la rappresentazione ASCII dei testi.

* BINARIO

1.9.3 Tavola dei settori di un file (Track/Sectors List)

Contiene gli indirizzi (numero di pista, numero di settore) dei settori di un file. Il suo formato è il seguente:

	Byte	Utilizzo
	0	Vuoto
	1	Pista } del prossimo settore di
	2	Settore (0-F) } memorizzazione della T/SL
	3, 4	Vuoto
	5, 6	Numero di settori utilizzati per memorizzare questa tavola T/SL
122 locazioni	da 7 a B	Vuoto
	da C, D	Numero della pista, numero del settore (0-F)
	da E, F	Numero della pista, numero del settore (0-F)
	da 10, 11	Numero della pista, numero del settore (0-F)
	da .	
	da :	
	da FE, FF	

Tutti i settori della T/S sono uguali. L'indirizzo del primo è contenuto nel catalogo.
Per i files di tipo testo, ad accesso diretto, che possono contenere numerosi buchi, è occupato solo lo spazio effettivamente necessario. Non è necessario riservare dello spazio. Se, per esempio, la lunghezza di un record è di 64 *bytes (*4 *records per settore) e volete scrivere il record numero* 2000, *sono riservati i seguenti settori:*

- 1 *settore per contenere il record* 2000;
- 5 *settori per la* T/S, *cioè*

$$\mathrm{INT}\left(\frac{2000}{122\times 4}+1\right)$$

ossia un totale di 6 *settori.*
Le locazioni corrispondenti ai records non ancora esistenti saranno messi a 0/0 (pista/settore) e aggiornati se il settore corrispondente è da creare.
Al contrario, se la dimensione di un file diminuisce, il DOS non libererà i settori non occupati.

1.9.4 Formato del catalogo di un mini floppy

Il DOS conserva le informazioni relative ai diversi files in un insieme di settori chiamato catalogo. Usando il comando CATALOG ottenete il loro contenuto. Su ogni disco inizializzato, la pista 17 ($11) è riservata al catalogo. Il catalogo inizia al settore $F della pista $11. I successivi settori (da $E a $1 della pista $11) gli vengono progressivamente assegnati man mano che gli occorrono. Dunque il catalogo può contenere al massimo 105 files.
Ogni settore del catalogo del disco ha il seguente formato:

Byte	Funzione
0	Vuoto
1	Pista } del seguito del catalogo
2	Settore } (0/0) se fine
da 3 a A	Vuoto
da B a 2D	Entrata del catalogo per il file 1
da 2E a 50	Entrata del catalogo per il file 2
da 51 a 73	Entrata del catalogo per il file 3
da 74 a 96	Entrata del catalogo per il file 4
da 97 a B9	Entrata del catalogo per il file 5
da BA a DC	Entrata del catalogo per il file 6
da DD a FF	Entrata del catalogo per il file 7

L'entrata di un file viene resa libera quando esso viene distrutto mettendo il primo byte dell'entrata in $FF.
Ciascuna entrata di file nel catalogo ha il seguente formato:

Byte	Funzione
0	Pista all'inizio della tavola dei settori del file (T/S List)
1	Settore
2	Tipo del file
	Bit 7 * file bloccato se il bit è = 1
	6
	5 future estensioni
	4
	3
	2 B file binario se questo bit è = 1
	1 A file APPLESOFT se questo bit è = 1
	0 I file Integer Basic se questo bit è = 1
	Tipo testo se i bits da 0 a 6 sono = 0
da 3 a 20	Nome del file
21	Numero di settori del file modulo 256

Se il numero è superiore a 256, la gestione del file dal DOS *non pone problemi tranne che per lo spazio libero indicato all'utente.*

1.9.5 Tavola dei contenuti di un mini floppy

È situata nel settore 0 della pista 17 ($11). Contiene le informazioni riportate nella tabella a pagina seguente.
Le tavole di settori liberi delle piste sono formate da 4 bytes. Gli ultimi due sono, per il DOS 3.3, posti a zero.

Byte num. esad.	Valore num. esad.	Descrizione
0	02	Vuoto
1	11	Pista del primo settore del catalogo
2	0F	Settore
3	04	Versione del DOS
4	00	Vuoto
5	00	Vuoto
6	da 1 a $FE	Volume del disco
da 7 a 26	00	Vuoto
30	$FF	Maschere per le tavole di settori liberi (bit a 1 = settore libero)
31	$FF	
32	$00	
33	$00	
34	$23	Numero di piste
35	$0F	Numero di settori
36	00	Numero di bytes per settore
37	01	
da 38 a 3B	00	Tavolta dei settori liberi delle piste da 0 a 2 (DOS)
da 3C a 3F	00	
da 40 a 43	00	
44, 45	?	Tavola dei settori liberi Pista $3
46, 47	0	
48, 49	?	Tavola dei settori liberi Pista 34
4A, 4B	0	
4C, 4D	?	Tavola dei settori liberi Pista $5
4E, EF	0	
...		
C1, C1	?	Tavola dei settori liberi Pista $22
C2, C3	0	
da C4 a FF	0	Vuoto

Ogni bit dei due primi bytes indica se il settore corrispondente è libero o meno.

- Bit = 1 settore libero;

- Bit = 0 settore occupato.

Un esempio di tavola è il seguente:

bit	0 0 0 0 1 1 1 1	1 1 1 1 1 1 1 1	0 0 0 0 0 0 0 0	0 0 0 0 0 0 0 0
settore	F E D C B A 9 8	7 6 5 4 3 2 1 0		

Le piste sono usate dal DOS nel seguente ordine:

da 18 a 34 (da $12 a $22) verso l'interno

poi

da 16 a 3 (da $10 a $03) verso l'esterno

I settori di una pista vengono allocati partendo da $OF e arrivando a $00.

1.9.6 Indirizzi di memoria del DOS su un Apple II 48 K

La seguente tabella fornisce gli indirizzi dove sono caricate le piste da 0 a 2 corrispondenti al DOS sul disco. Al momento del caricamento di un mini floppy master, il DOS viene caricato ad un indirizzo che dipende dalla dimensione della memoria. In un mini floppy secondario, il DOS viene caricato all'indirizzo in cui si trovava al momento dell'inizializzazione del disco.

Disco (principale o secondario)		**Memoria**	
Pista	**Settore**	**indirizzo di caricamento**	**indirizzo di spostamento**
00	00	$3600	$B600
00	01	$3700	$B700
00	02	$3800	$B800
00	03	$3900	$B900
00	04	$3A00	$Ba00
00	05	$3B00	$BB00
00	06	$3C00	$BC00
00	07	$3D00	$BD00
00	08	$3E00	$BE00
00	09	$3F00	$BF00
00	0A	$1B00	
00	0B	$1C00	
00	0C	$1D00	$9D00
00	0D	$1E00	$9E00
00	0E	$1F00	$9F00
00	0F	$2000	$A000
01	00	$2100	$A100
01	01	$2200	$A200
01	02	$2300	$A300
01	03	$2400	$A400
01	04	$2500	$A500

(segue)

(seguito)

Disco (principale o secondario)		Memoria	
Pista	Settore	indirizzo di caricamento	indirizzo di spostamento
01	05	$2600	$A600
01	06	$2700	$A700
01	07	$2800	$A800
01	08	$2900	$A900
01	09	$2A00	$AA00
01	0A	$2B00	$AB00
01	0B	$2C00	$AC00
01	0C	$2D00	$AD00
01	0D	$2E00	$AE00
01	0E	$2F00	$AF00
01	0F	$3000	$B000
02	00	$3100	$B100
02	01	$3200	$B200
02	02	$3300	$B300
02	03	$3400	$B400
02	04	$3500	$B500

Su un APPLE II 48 K, il DOS occupa dall'indirizzo $9600 al $BFFF (fine della memoria RAM), ossia dal 38400 al 49151, cioè circa 10 K.
Occupa anche alcuni bytes nella pagina tre della memoria.

Vettori del DOS			
3D0	JMP	$9DBF	Ricaricamento "a caldo" del DOS
3D3	JMP	$9D84	Ricaricamento completo del DOS
3D6	JMP	$AAFD	Salto al gestore di files
3D9	JMP	$B7B5	Salto a RWTS
3DC	LDA	$9D0F	Ripristino dei parametri per il gestore di files
	LDY	$9D0E	(indirizzo della tavola FMPL)
	RTS		
3E3	LDA	$AAC2	Ripristino dei parametri per RWTS
	LDY	$AAC1	(indirizzo della tavola IOB)
	RTS		
3EA	JMP	$A851	Aggiornamento del DOS per i buffer di I/O
	NOP		
	NOP		

Vettori del monitor e dell'APPLESOFT.
Questo paragrafo descrive il sottoprogramma RWTS (accesso diretto ai dischi senza passare dal DOS) e il gestore dei files.

Abbiamo indicato all'indirizzo $3EA un salto ad un sottoprogramma di aggiornamento del DOS per i buffers di input/output. Cosa succede quando digitate PRINT "COUCOU" nel momento in cui il DOS è attivato, ma non sta accedendo ad un file (cioè non ci sono istruzioni READ o WRITE in atto)?
I registri del monitor CSWL,H e KSWL,H (cf. volume 1) contengono l'indirizzo del DOS; questo permette al monitor di trasmettergli l'istruzione di input/output. Il DOS, avendo conservato gli indirizzi delle unità di controllo di input/output in alcuni registri interni, lancia l'input/output. Alla fine di questa operazione, il DOS rende il controllo al monitor riportando il suo indirizzo in CSWL,H e KSWL,H. Questo è il ruolo del sottoprogramma il cui vettore si trova in $03EA.

1.9.7 Accesso diretto ai settori fisici del disco

Nell'APPLE II 48 K, il sottoprogramma RWTS (Read Write Track Sector = leggere scrivere pista settore) si trova all'indirizzo $B7B5. È più semplice richiamare sempre il vettore RWTS in $3D9 che è indipendente dalla dimensione della memoria e che contiene un salto a RWTS.
I parametri di chiamata del RWTS sono due tabelle chiamate IOB (blocco di controllo di Input/Output) e DCT (caratteristiche del drive). Il sottoprogramma $3E3 fornisce nei registri A (byte alto) e Y (byte basso) l'indirizzo della tabella IOB in memoria.
Analizziamo più in dettaglio queste due tabelle.

• DCT

Ha il seguente formato:

Tipo	Fase	Motore ON
$00	$01	$EFD8

Sono le caratteristiche del drive. Esse sarebbero differenti, per esempio, con un disco non flessibile.

• IOB

Byte	Nome	Funzione
1	IBTYPE	
2	IBSLOT	Numero dello slot dell'unità dei controllo *16
3	IBDRVN	Numero del drive ($1 o $2)
4	IBVOL	Volume del disco al quale si dovrà accedere
5	IBTRK	Pista } ai quali accedere
6	IBSEC	Settore } ai quali accedere

(segue)

(seguito)

Byte	Nome	Funzione
7 e 8	IBDCTP	Indirizzo della tavola DCT
9 e 10	IBBUFP	Indirizzo del buffer di 256 bytes dove leggere (o scrivere) i dati da scrivere (o lettere) mediante RWTS
11 e 12	vuoto	
13	IBCMD	Codice del comando per RWTS $00 POSIZIONAMENTO DELLA TESTINA $01 LETTURA $02 SCRITTURA $04 FORMATTAMENTO DEL DISCO
14	IBSTAT	Codice di errore se il bit di riporto (cfr. par. 1.2) è = 1 $08 errore di formattamento $10 protezione in scrittura $20 volume errato $40 errore di disco $50 errore di lettura
15	IBSMOD	Volume letto sul disco
16	IOBPSN	Ultimo slot
17	IOBPDN	Ultimo drive al quale si è avuto accesso

1.9.8 Gestore di files

Il DOS è costituito, oltre che dal programma RWTS di accesso ai dischi, da un interprete di comandi e dal gestore di files. L'interprete di comandi lavora sui dati presi da tastiera in modo diretto o comunicati dal Basic (CTRL-D) tramite programma. Per ogni comando che lavora su files, prepara una tavola di dati chiamata FMPL il cui indirizzo può essere ricavato chiamando il sottoprogramma di indirizzo $3DC (cf. par. 1.9.6). Questo fornisce nell'accumulatore il byte alto e nel registro di indice Y il byte basso di quest'indirizzo.
La tavola può contenere fino a 18 bytes la cui descrizione dipende dal comando da realizzare, seguendò il formato della pagina.
I possibili codici di ritorno sono (+ 0A):

Codice	Significato
00	Senza errori
04	Protezione in scrittura
05	Fine file
06	File inesistente
07	Errore di volume
08	Errore fisico
0A	Files bloccati

La chiamata del gestore di files si esegue con le istruzioni:

```
LDX #0;       creazione del file autorizzato
JSR $3D6
BCS errore;   salto all'errore
```

1.9.9 Indirizzi utili del DOS 3.3

- **File binario**

Dopo aver digitato il comando BLOAD FICH1, è utile conoscere l'indirizzo dei caricamento e la lunghezza del file.
Nell'APPLE II 48 K, l'indirizzo di caricamento è contenuto in $AA72, AA73 e la lunghezza in $AA60.

- **Tavola dei comandi del DOS**

Il DOS riconosce i comandi grazie ad una tavola situata all'indirizzo $AA88 che contiene i codici ASCII dei nomi dei comandi. Ognuno dei 28 comandi è separato dal successivo mediante il bit alto del suo ultimo carattere, che è 1. La lunghezza totale della tavola è al massimo 133 caratteri e non può contenere spazi.
L'ordine dei comandi del DOS nella tavola:

INIT, LOAD, SAVE, RUN, CHAIN, DELETE, LOCK,
UNLOCK, CLOSE, READ, EXEC, WRITE, POSITION,
OPEN, APPEND, RENAME, CATALOG, MON, NOMON,
PR, IN, MAXFILES, FP, INT, BSAVE, BLOAD, BRUN,
VERIFY

Potete anche personalizzare il vostro DOS e, ad esempio, sostituire il comando CATALOG con CAT più veloce da battere. Una volta fissata la vostra tavola, dovrete riregistrare il vostro DOS su disco usando ad esempio il programma del paragrafo 1.9.10.

- **Sostituzione del messaggio DISK VOLUME**

Se analizzate il contenuto degli indirizzo compresi tra $B3B0 e $B3BA ottenete:

```
B3B0 C5 CD D5 CC CF D6 A0 CB
B3B8 D3 C9 C4
```

Questi codici corrispondono ai seguenti codici ASCII APPLE:

```
B3B0 E M U L O V K
B3B8 S [I] D
```

Per personalizzare i vostri dischi potete benissimo sostituire questo messaggio DISK VOLUME con un altro. Per esempio, mettete il vostro nome e usate il volume per numerare i vostri dischi.

	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10	11
OPEN	01		Lunghezza di un campo o 0000		V	D	S	Tipo del file	Indirizzo del buffer del file		Codice di resoconto (cfr. sotto)		Indirizzo della zona di lavoro gestionale dei files (buffer da 45 bytes)		Indirizzo della Track/Sector List			
CLOSE	02																Indirizzo del buffer dei dati	
READ	03		Numero del campo		Numero di byte		Lunghezza dei dati		Indirizzo relativo dei dati nel buffer									
WRITE	04																	
DELETE	05				V	D	S		Indirizzo del buffer del file									
CATALOG	06				V	D	S											
LOCK	07				V	D	S		Indirizzo del buffer del DOS relativo al (nome del) file						Indirizzo della Track/Sector List			
UNLOCK	08				V	D	S											
RENAME	09		Indirizzo del buffer del nuovo nome		V	D	S											
POSITION	0A		N° del campo		N° byte													
INIT	0B				V	D	S											
VERIFY	0C				V	D	S		come sopra						Indirizzo della T/SA List		Indirizzo buffer dei dati	

V volume, D drive, D slot

Tavola **FMPL** dei dati del gestore dei files.

Nome del file		
		ABTGF
ATSL	ADATA	ABS

ABTGF indirizzo del buffer di lavoro del gestore dei files
ATSL indirizzo della tavola dei settori liberi
ADATA indirizzo del buffer dei dati del settore (256 bytes)
ABS indirizzo del nome file seguente

Formato del buffer di un file del DOS

Inoltre, è possibile usare i vari modi video dello schermo per i caratteri del messaggio.

Se	L = ASC(L$)	
e	L < 64	il codice schermo del carattere è: C + 128 modo normale C + 64 modo lampeggiante C modo inverso
e	L > 64	il codice schermo del carattere è: C + 128 modo normale C modo lampeggiante C— 64 modo inverso

La lunghezza del vostro messaggio è limitata a 11 *caratteri e dovete invertirlo in memoria.*
Per conservare la modifica, dovete rimemorizzare il vostro DOS *su disco* (cfr. par. 1.9.10).

• **Parametri del sistema**

Se volete conoscere, in un dato insieme, le caratteristiche del sistema, potete esaminare i contenuti dei seguenti indirizzi:

$AA6A	—21910	slot attuale
$AA68	—21912	drive (1 o 2) attivo
$AA66	—21914	volume attuale
$AA6C	—21908	lunghezza di campo
$AA6E	—21906	numero di campo
$AA70	—21904	numero del byte attuale

1.9.10 Copiatura di un DOS modificato su disco

Se digitate il seguente programma:

```
300: A909    LDA  $09
     8D5DAA  STA  $AA5D
     A900    LDA  $00
     8D5FAA  STA  $AA5F
     A99D    LDA  $9D
     8DBCB5  STA  $B5BC
     20C2B7  JSR  $B7C2
     204AB7  JSR  $B74A
     60      RTS
```

e in seguito battete 300G, il DOS caricato in memoria in RAM verrà ricopiato sulle piste 0, 1 e 2 del vostro disco.

1.10 TRATTAMENTO DEGLI ERRORI

Viene effettuato come con l'APPLESOFT. Sono utilizzati nuovi codici d'errore:

Codice	Messaggio	Significato
1	LANGUAGE NOT AVAILABLE	Linguaggio non disponibile
2, 3	RANGE ERROR	Il parametro di un comando è al di fuori dell'intervallo concesso
4	WRITE PROTECTED	Disco protetto in scrittura (cfr. par. 1.2)
5	END OF DATA	Fine file (sono stati letti tutti i dati)
6	FILE NOT FOUND	File non trovato
7	VOLUME MISMATCH	Errore di volume
8	I/O ERROR	Errore fisico (disco non inizializzato, sportello aperto...)
9	DISK FULL	Disco pieno
10	FILE LOCKED	File bloccato (protetto in scrittura)
11	SYNTAX ERROR	Errore di sintassi
12	NO BUFFERS AVAILABLE	Troppi file di tipo TEXT sono aperti (cfr. par. 1.6.2 MAXFILES)
13	FILE TYPE MISMATCH	Tipo file errato
14	PROGRAM TOO LARGE	Dimensione di un programma troppo grande
15	NOT DIRECT COMMAND	Comando non utilizzabile in modo diretto

CAPITOLO 2

Il sistema U.C.S.D.

2.1 PRESENTAZIONE

2.1.1 Generalità

Il sistema U.C.S.D. consente di utilizzare sull'APPLE II dei linguaggi evoluti come Pascal, Fortran, ... e fornisce assieme ad essi un assembler evoluto.
Sull'APPLE IIe, per disporre del sistema U.C.S.D., è sufficiente avere i dischi corrispondenti, mentre con l'APPLE II +, è necessaria una scheda di espansione 16 K, la scheda dei linguaggi di programmazione.

2.1.2 Struttura in memoria della scheda dei linguaggi

La scheda dei linguaggi vi offre 16 K bytes di memoria RAM supplementari, per un totale di 64 K bytes di RAM alla capacità di memoria indirizzabile del microprocessore 6502. La RAM fornita può essere in parte protetta in scrittura; in essa memorizzerete i vostri dati più importanti.
La scheda dei linguaggi include, inoltre, la ROM Autostart. Essa può funzionare con le diverse versioni del DOS 3.2, 3.3...
La scheda di memoria dell'APPLE II + diventa la seguente:

$FFFF **$F800**	**Scheda dei linguaggi**	**ROM Autostart** **scheda dei linguaggi**
$ F7FF $E000	8 K RAM supplementari	ROM APPLESOFT o Basic

(segue)

(seguito)

$FFFF **$F800**	**Scheda dei linguaggi**		**ROM Autostart scheda dei linguaggi**
$DFFF	1° blocco 4 K RAM	2° blocco 4 K RAM	Intero
$D000			
$CFFF	Input/Output schede di espansione		
$C000			
	RAM 48 K bytes		
$0000			

Per rendere 64 K *bytes di* RAM *realmente accessibili all'utente, occorre sostituire quella parte di memoria riservata agli Input/Output con della memoria* RAM. *Essendo complicato,* APPLE *ha preferito fornire nella scheda dei linguaggi due blocchi da* 4 K RAM *selezionabili separatamente agli stessi indirizzi.*

Tabella delle possibilità di selezione, di accesso in lettura, in scrittura e di protezione in scrittura della memoria.

RAM $D000-FFFF		**Selezione RAM/ROM e protezione in scrittura RAM**
1° blocco 4 K incluso	2° blocco 4 K incluso	
$C080	$C088	Selezione in lettura RAM
—16256	—16248	Protezione in scrittura RAM
$C081	$C089	Selezione in lettura ROM
—16255	—16247	Senza selezione in lettura RAM Selezione in scrittura RAM dopo due letture di questo indirizzo
$C082	$C08A	Selezione in lettura ROM
—16254	—16246	Senza selezione in lettura RAM Protezione in scrittura RAM
$C083	$C08B	Selezione in lettura RAM
—16253	—16245	Selezione in scrittura RAM dopo due letture di questo indirizzo

Gli indirizzi

{ $C080-C083 e $C084-$C087
{ $C088-C08B e $C08C-$C08F

sono equivalenti in quanto non si prende in considerazione il bit 2 dell'indirizzo.

2.1.3 Utilizzazione con il DOS 3.3

La scheda dei linguaggi può essere utilizzata in uno dei seguenti due modi:

{ RAM Integer Basic
{ ROM APPLESOFT

oppure

{ RAM APPLESOFT
{ ROM Integer Basic

I comandi FP, INT diventano facili da usare (cfr. cap. 1).

2.2 MESSA IN FUNZIONE DEL SISTEMA

2.2.1 Introduzione

In questo paragrafo realizzeremo un esempio di programma Pascal, spiegando ciò che occorre per far funzionare il sistema U.C.S.D., il sistema di gestione dei files, il text editor, il compilatore Pascal.

2.2.2 Avviamento del sistema P

• APPLE II monodrive

Il caricamento del sistema si esegue in due tappe:

- inserite il disco APPLE 3 nel vostro drive, accendete il vostro APPLE e chiudete lo sportello dell'unità disco;
- inserite a questo punto, il disco APPLE 0 e battete Reset. Compare il seguente messaggio:

```
WELCOME APPLE 0, TO
U.C.S.D. PASCAL SYSTEM II.I
CURRENT DATE IS 28-FEB-82
```

In cima allo schermo leggete:

```
COMMAND: E(DIT, R(UN, F(ILE, C(OMP, L(IN
```

A questo punto potete editare un file, compilare un programma, ecc.

- **APPLE II pluridrive**

È sufficiente una sola tappa. Inserite il disco APPLE 1 nell'unità disco 1 dello slot 6. Avviate il vostro APPLE II. Leggete

WELCOME APPLE1, TO
U.C.S.D. PASCAL SYSTEM II.I
CURRENT DATE IS 30-JAN-82

A questo punto potete editare un file, ..., ma non compilare un programma. Per poterlo fare inserite il disco APPLE 2 in un altro drive. Infatti, il disco APPLE 1 non contiene il compilatore Pascal.

2.2.3 Modifica della data

Prendete la buona abitudine di cambiare la data al caricamento del sistema. Per farlo digitate il comando F(ILE e poi D, vedrete apparire il messaggio:

DATE SET: <1..31>-<JAN..DEC>-<00..99>
TODAY IS 28-FEB-82
NEW DATE?

Digitate la nuova data (giorno, mese, anno), poi battete Q (ritorno al livello comando). La linea

COMMAND: E(DIT, R(UN, F(ILE, C(OMP, L(IN

riappare in cima allo schermo.

2.2.4 Formattamento di nuovi mini floppy

Per poter cominciare a lavorare sul vostro sistema, vi occorrono altri dischi oltre agli APPLE 0:, 1:, 2:, 3: che sono protetti in scrittura. Per utilizzare dei dischi vergini, dovete formattarli (inizializzarli). Il file APPLE3:FORMATTER deve essere inserito nel drive. In seguito digitate il comando X (esecuzione di un programma). Il sistema risponde:

EXECUTE WHAT FILE?

scrivete:

APPLE3:FORMATTER

Il seguente messaggio compare sullo schermo:

APPLE DISK FORMATTER PROGRAM
FORMAT WHICH DISK (4, 5, 9, 12)?

Togliete dai drives tutti i dischi non vergini, per non rischiare di perdere i loro contenuti. Preparate una pila di dischi vergini. Inserite ora un disco vergine al posto del disco di caricamento (APPLE0: o 1:) e digitate 4. Il drive si mette in funzione e compare il seguente messaggio:

NOW FORMATTING DISKETTE IN DRIVE 4

Quando l'operazione termina, il programma pone la domanda:

FORMAT WHICH DISK (4, 5, 9..12)?

Se volete formattare un altro disco, inseritelo nello stesso drive e ribattete 4. Altrimenti inserite il vostro disco di sistema e battete Return. Il sistema ritorna al livello comando (COMMAND:).
Se il programma riconosce un disco Pascal già formattato, vi chiederà la conferma visualizzando il messaggio:

DESTROY DIRECTORY OF nome del disco?

Battete N e il vostro disco sarà conservato, Y *e sarà reinizializzato.*

2.2.5 Copia di dischi

Per evitare dei disastri e la perdita del sistema Pascal, vi consigliamo vivamente di copiare i quattro dischi APPLE (0:, 1:, 2:, 3:) e di usare solo le copie.
Per copiare un disco, portatevi al livello Filer digitando il comando F(ILE. Comparirà la seguente linea:

FILER:G, S, N, L, R, C, T, D, Q

Inserite il disco da copiare in un drive e, se avete un APPLE II pluridrives, il disco di destinazione in un altro drive. Digitate poi il comando T.
Compare la seguente domanda:

TRANSFER?

Digitate il nome del disco da copiare e Return. Il sistema verifica che il nome corrisponda ad un disco nel drive e poi fa comparire:

TO WHERE?

Indicate il nome del disco di destinazione. Il sistema vi chiederà conferma con:

TRANSFER 280 BLOCKS? (Y/N)

Se vi siete sbagliati digitate N e il comando sarà ignorato, digitate Y e viene lanciata la copia.

Esempio:

TRANSFER? APPLE0:
TO WHERE? BLANK: (nome di un disco vergine formattato)
TRANSFER 280 BLOCKS? (Y/N) Y

I due punti (:) *nel nome di un disco sono molto importanti in quanto differenziano i nomi dei files dai nomi dei dischi.*

Il sistema ricerca, in seguito, il disco BLANK:. Se lo trova chiede:

DESTROY BLANK:?

Rispondete Y (sì) e BLANK: diventerà la copia perfetta del disco sorgente.
Se il Filer non trova BLANK:, domanda:

PUT IN BLANK:
TYPE <SPACE> TO CONTINUE

Inserite BLANK: in un drive (per esempio togliendo APPLE0:) e battete uno spazio.
Nel caso abbiate un sistema monodrive, il sistema specificherà il disco da inserire nel vostro drive. Una volta terminata la copia, il FILER visualizzerà:

APPLE0:→BLANK:

Dopodiché battete Q(uit per uscire dal Filer e ritrovarvi al livello comando.

2.2.6 Creazione di un programma

Scriviamo ora un programma. Al livello comando, selezioniamo il text editor digitando E. Compare il seguente messaggio:

>EDIT:
NO WORKFILE IS PRESENT. FILE? (<RET> FOR NO FILE <ESC-RET> TO EXIT)

In realtà vedete solo i primi 40 caratteri del messaggio. Per leggere gli altri, battete CTRL-A, e per ritornare, battetene un altro. Questo è dovuto al fatto che il Pascal gestisce uno schermo di 80 colonne mentre lo schermo dell'APPLE ha solo 40 colonne.
La parola EDIT significa che il text editor è stato attivato. Noi dobbiamo creare un nuovo file. Battiamo RETURN (<RET> FOR NO FILE). La linea dei comandi dell'editor compare in cima allo schermo.

>EDIT: A(DJST C(PY D(LTE F(IND I(NSRT J(MP
R(PLACE Q(UIT X(CHNG Z(AP

La descrizione dettagliata di questi comandi si trova nel paragrafo 2.5. Qui descriviamo solo in parte i comandi I (inserimento di testi), D (soppressione di testi) e Q (uscita dell'editor).
Battiamo I per prendere il nostro programma:

>INSERT: TEXT <BS> A CHAR, <DEL> A LINE
<EXT> ACCEPTS, <ESC> ESCAPS

A partire da questo momento, tutto ciò che prenderete apparirà sullo schermo. Quando avete finito, battete CTRL-C(EXT) e il vostro inserimento sarà definitivo. Se vi sbagliate, potete tornare indietro con ←, o cancellare tutti i caratteri presi con ESC.
Per cancellare dei caratteri errati, dopo aver digitato ETX, dovrete utilizzare un altro comando D. Posizionate il cursore sul primo carattere da cancellare sui caratteri da cancellare.
Quando avete finito, battete CTRL-C: comparirà la nuova linea corretta.

Se avete dimenticato un carattere, dovete usare nuovamente il comando I. Posizionate il cursore sul carattere davanti al quale deve essere inserito il testo.

Esempio:

PROGAM

mettete il cursore sulla A, battete I R CTRL-C e la linea diventerà

PROGRAM

Il programma ARAROM, scritto in Pascal, effettua la conversione di un numero arabo nel corrispondente numero romano.

```
PROGRAM ARAROM;
CONST
        LGMAX = 18;
TYPE
        STRINGA = RECORD
                        CARATTERI  : ARRAY[1..LGMAX] OF CHAR;
                        LUNGHEZZA  : 0..LGMAX
                  END ;
        NUMERARA = 0..3999;
VAR
        X : NUMERARA;
        NUMROM : STRINGA;
        I :1..LGMAX;

FUNCTION CONVAR (X:NUMERARA):CHAR;
BEGIN (* CONVAR *)
CASE X OF
             1:CONVAR:='I';
             5:CONVAR:='V';
            10:CONVAR:='X';
            50:CONVAR:='L';
           100:CONVAR:='C';
           500:CONVAR:='D';

END (* DEL CASE *);

IF X=1000 THEN CONVAR:='M'

END; (* CONVAR *)

FUNCTION BASE(X:NUMERARA):NUMERARA;

(* QUESTA FUNZIONE CERCA IL MASSIMO NUMERO
   CORRISPONDENTE AD UNA CIFRA ROMANA INFERIORE
   A X *)
```

```
BEGIN (* BASE *)

IF X>=1000 THEN BASE:=1000
ELSE IF X>=500 THEN BASE:=500
     ELSE IF X>=100 THEN BASE:=100
          ELSE IF X>=50 THEN BASE:=50
               ELSE IF X>=10 THEN BASE:=10
                    ELSE IF X>=5 THEN BASE:=5
                         ELSE BASE:=1
END; (* BASE *)

PROCEDURE CONCAT (C:STRINGA;VAR CH:STRINGA);

(* QUESTA PROCEDURA CONCATENA UN CARATTERE
   IN CODA AD UNA STRINGA DI CARATTERI *)

BEGIN (* CONCAT *)
CH.LUNGHEZZA := CH.LUNGHEZZA + C.LUNGHEZZA ;
FOR I:= LGMAX DOWNTO 1 DO
BEGIN
IF I>C.LUNGHEZZA THEN
                    CH.CARATTERI[I] := CH.CARATTERI[I-C.LUNGHEZZA]
                ELSE
                    CH.CARATTERI[I] := C.CARATTERI[I]
END;
END; (* CONCAT *)

PROCEDURE TRASFORM (X:NIMERARA;VAR CH:STRINGA);
VAR
        C : STRINGA;

BEGIN (* TRASFORM *)
C.LUNGHEZZA := 1;
CH.LUNGHEZZA := 0;
WHILE X<>0 DO
           BEGIN
           C.CARATTERI[C.LUNGHEZZA] :=CONVAR(BASE(X));
           CONCAT(C,CH);
           X := X-BASE(X);
           END;

END (* TRASFORM *)

BEGIN (* ARAROM *)
WHILE NOT EOF DO BEGIN
    FOR I:=1 TO LGMAX DO NUMROM.CARATTERI[I]:=' ';
    READLN(X);
    WHILE (X>=4000) DO READLN(X);
    TRASFORM(X,NUMROM);
```

```
    WRITE ('IL NUMERO ROMANO CORRISPONDENTE AL NUMERO ',X,' E` IL
                                                          SEGUENTE ');
    FOR I:=LGMAX DOWNTO 1
        DO WRITE (NUMROM.CARATTERI[I]);
    WRITELN;
END;
END. (* ARAROM *)
```

Adesso memorizzate questo programma. Quando siete al livello comando dell'editor, battete Q. Comparirà il seguente messaggio

>QUIT:

> U(PDATE THE WORKFILE AND LEAVE
> E(XIT WITHOUT UPDATING
> R(ETURN TO THE EDITOR WITHOUT UPDATING
> W(RITE TO A FILE NAME AND RETURN
> S(AVE WITH SAME NAME AND RETURN

Battere V per creare un "file di lavoro" SYSTEM, WRK, TEXT.
Il sistema visualizza:

> WRITING...
> YOUR FILE IS xxx BYTES LONG

Prima di eseguire il programma, occorre compilarlo. Il comando R lancia la compilazione, il collegamento, se necessario, e l'esecuzione. Leggerete:

> RUNNING...

Se, durante la compilazione, viene trovato un errore, il sistema fa comparire un messaggio simile a questo:

PROGRAM < < < <
LINE 1, ERROR 18: <SP> (CONTINUE), <ESC> (TERMINATE),
E(DIT

Battete E e ritornerete automaticamente nell'editor.

2.3 IL LIVELLO COMANDO

2.3.1 Generalità

Dopo aver caricato sul vostro APPLE il sistema Pascal, avete visto comparire sullo schermo la seguente linea:

COMMAND: E(DIT, R(UN, F(ILE, C(OMP, L(INK, X(ECUTE,
A(SSEM, D(EBUG,?

o metà linea se lo schermo ha solo 40 caratteri. L'altra metà viene visualizzata battendo CTRL-A (cfr. par. 2.1).
Questa linea è standard nel Pascal UCSD (sistema P). Tuttavia sull'APPLE alcune funzioni non sono attivate (D(EBUG); se digitale D il comando viene ignorato.
Descriveremo brevemente in questo paragrafo, i diversi comandi e i files necessari per metterli in funzione.

2.3.2 Breve semantica dei vari comandi

La linea COMMAND qui sotto non fornisce l'intera lista dei comandi. Infatti la lista completa non può essere contenuta in una linea di 80 caratteri. Questa è la ragione per cui compare un punto di domanda ?. Il resto della linea è ottenuto battendo un punto di domanda sulla tastiera.

COMMAND: U(SER RESTART, I(NITIALIZE, H(ALT

- **F(ILE**

Battere F(ILE quando si è al livello comando, porta l'utente ad essere inserito nel sistema di gestione dei files. Potete memorizzare, distruggere, leggere, trasferire files e il file di lavoro, consultare il catalogo dei dischi, verificarli fisicamente, fissare definitivamente i settori che non vanno, ... (cfr. par. 2.4 *Il sistema di gestione dei files*).

- **E(DIT**

Avete già visto che battere E(DIT quando si è al livello comando porta l'utente nell'editor. Potete inserire testi in un file, cancellarne, modificarne, ecc. (cfr. par. 2.5 *Il text editor*).

- **C(OMPILE**

Battere C, quando si è al livello comando, lancia il compilatore. Esso controlla se esiste un file di lavoro. Se esiste lo compila, altrimenti vi chiede il nome del file da compilare. Quando trova un errore, il compilatore lo visualizza e chiede all'utente se desidera modificare il file sorgente e ritornare nell'editor, oppure continuare la compilazione. Quando si arriva ad una compilazione senza errori, viene generato un file oggetto in P-code.

- **A(SSEMBLE**

Il sistema Pascal offre all'utente la possibilità di creare dei sotto-programmi assembler 6502 chiamati dal Pascal. Per lanciare l'assemblaggio, dovete battere A quando siete al livello comando. Come il compilatore, l'assemblatore vi chiede, quando trova un errore, se volete ritornare nell'editor oppure continuare l'assemblaggio (cfr. par. 2.6 *Assemblatore*).

- **L(INK**

Questa opzione permette all'utente di creare un programma eseguibile a partire dal suo file compilato, da sotto-programmi assembler o Pascal

compilati separatamente e che si trovano in una biblioteca (cfr. par. 2.7 *Collegatore*).

• X(ECUTE

Per eseguire un programma che è stato compilato, battete X al livello comando. Il sistema vi chiederà quale programma volete eseguire.

EXECUTE WHAT FILE?

Dovete specificare il nome di un file. Il sistema aggiunge .CODE al nome e cerca il file sul disco. Se desiderate eseguire un programma il cui nome non termina con .CODE, date come nome del file il nome del vostro programma seguito da un punto.
Per esempio, è uguale indicare come nome di file:

ROMARA

e

ROMARA.CODE

Il sistema lancia l'esecuzione del vostro programma. Nei casi seguenti, però, questo non è possibile:

- Il programma non è stato "linkato" (sono rimaste delle etichette non definite). Il sistema visualizza:

MUST L(INK FIRT

e torna al livello comando. Potete digitare L(INK (cfr. qui sopra).
– Il file specificato non contiene P-code (file di testo, file di dati...).
Leggerete:

nome del file NOT CODE

- La biblioteca di sottoprogrammi SYSTEM.LIBRARY non è sul disco. Comparirà il messaggio:

REQUIRED INSTRINSIC(S) NOT AVAILABLE

Questo indica che mancano dei sottoprogrammi che generalmente si trovano nella biblioteca (grafica, gestione delle manopole per giochi, Input/Output...).
- Il programma utilizza i sottoprogrammi WCHAR o WSTRING (cfr. *TURTLEGRAPHICS*, Volume 3 Pascal UCSD) e il file SYSTEM. CHARSET non è sul disco. Il programma viene eseguito ma non compaiono messaggi sullo schermo.

Il comando X(ECUTE consente di eseguire dei programmi corretti senza doverli ricompilare od ogni esecuzione.

• R(UN

Il comando R(UN lancia la compilazione del file di lavoro, il collegamento, se necessario, e, se è possibile, l'esecuzione.

Durante l'esecuzione del comando R(UN, *il disco del sistema di caricamento deve restare nel drive, in quanto tra la compilazione, il collegamento e l'esecuzione, c'è un attimo in cui si torna al livello comando.*
Il comando R(UN è molto utile durante la messa a punto di un programma, la quale si effettua usando alternativamente i comandi E(DIT e R(UN.

• D(EBUG

Questo comando fa parte del livello comando standard Pascal, ma non è implementato sull'APPLE II. Tuttavia, se battete D al livello comando, viene lanciato il compilatore.

• U(SER RESTART

Questo comando rilancia l'ultimo comando utilizzato. Per esempio, se digitate i comandi:

X(ECUTE

e

U(SER RESTART

il programma verrà eseguito due volte.

• I(NITIALIZE

Battendo I viene reinizializzato il sistema, ma non viene ricaricato in memoria l'interprete P-code. Il disco di caricamento (Boot) deve essere presente. In pratica viene rilanciata la seconda fase di caricamento del sistema P (cfr. *Caricamento con un solo drive*).
Quando c'è un errore nello svolgimento di un programma (divisione per zero, overflow...), questo comando viene eseguito automaticamente.

• H(ALT

Battere H(ALT equivale a spegnere e accendere il vostro APPLE. L'interprete P-code viene ricaricato in memoria e occorre ripetere tutta la procedura di avviamento del sistema P.

2.3.3 Dischi forniti

• Files necessari ai comandi del sistema

Il sistema Pascal è un programma troppo grosso per essere conservato ininterrottamente in memoria. In un dato istante utilizzate solo una piccola parte del sistema. Sul disco, esso è scomposto in files e, a seconda del comando desiderato, viene caricato in memoria il file necessario. I files corrispondenti ad un comando possono trovarsi su qualunque disco (il sistema, per trovarlo, percorre tutti i dischi) tranne che sul file di lavoro (Workfile SYSTEM.WRK.TEXT e SYSTEM.SYNTAX).

1. *È sempre preferibile verificare che i files necessari ad un comando siano presenti sui dischi utilizzati. Altrimenti il sistema si "fermerà" e sarete costretti a ricaricarlo.*

2. *L'accesso ai files* SYSTEM.PASCAL *e* SYSTEM.LIBRARY *si effettua diversamente dagli altri files. Al caricamento, il sistema rileva la loro posizione; quando dovrà utilizzarli, userà queste posizioni senza riconsultare il contenuto dei dischi. Se avete spostato i vostri dischi, il sistema verrà perso e si fermerà; dovrete allora ricaricarlo. Per evitare questo problema, quando spostate i dischi digitate il comando* I(NITIALIZE.

Diamo qui sotto l'elenco dei files necessari alle diverse parti del sistema.

Parte del sistema	**Files necessari**	**Dischi in cui si deve trovare il file**
Livello comando	SYSTEM.PASCAL	Questo file deve trovarsi dove si trovava al momento del caricamento del sistema. Se il disco di caricamento è presente e se il file SYSTEM.PASCAL non si trova più allo stesso posto, il sistema si blocca e occorre ricaricarlo. Se il disco di caricamento è assente, il sistema chiede di reinserirlo.
F(ILE	SYSTEM.FILER Files utilizzati	Presenza su un disco qualunque (numero unità di controllo e drive indifferenti). File di lavoro per default (situato sul disco di caricamento)
E(DIT	SYSTEM.EDITOR File sorgente da editare	Presenza su un disco qualunque (numero unità di controllo e drive indifferenti). File di lavoro per default (situato sul disco di caricamento)
C(OMPILE	SYSTEM.COMPILER	Qualunque disco tranne LIBRARY
	SYSTEM.LIBRARY	Usato se necessario
	SYSTEM.EDITOR	Usato per correggere gli errori
	SYSTEM.SYNTAX	Messaggi d'errore quando si correggono gli errori
	File sorgente da compilare	Disco qualunque file di lavoro per default (sul disco di caricamento)
A(SSEMBLE	SYSTEM. ASSEMBLER	Qualunque disco
	6500.OPCODES	Codici 6502. File sempre necessario
	6500.ERRORS	Messaggi d'errore di assemblaggio
	File sorgente da assemblare	Disco qualunque file di lavoro per default sul disco di caricamento
	SYSTEM.EDITOR	Per correggere gli errori

(segue)

(seguito)

Parte del sistema	Files necessari	Dischi in cui si deve trovare il file
L(INK	SYSTEM.LINKER	
	File da "linkare"	Per default SYSTEM.WRK.CODE situato sul disco di caricamento
	File-libreria	Per default SYSTEM.LIBRARY
X(ECUTE	File da eseguire	Necessario al momento del caricamento
	SYSTEM.LIBRARY	Necessario se il programma usa i sottoprogrammi della biblioteca
	SYSTEM.CHARSET	Necessario se vengono utilizzati WCHAR e WSTRING
R(UN	File sorgente in oggetto	Qualunque disco. Per default file di lavoro WRK sul disco di caricamento
	SYSTEM.COMPILER	Necessario se file di testo
	SYSTEM.EDITOR	Correzione degli errori di compilazione
	SYSTEM.SYNTAX	Messaggi di errore di compilazione
	SYSTEM.LINKER	Necessario se il programma usa dei sottoprogrammi esterni
	SYSTEM.LIBRARY	Necessario col collegamento (I/O files, grafica, giochi,...)
	SYSTEM.PASCAL	Ritorno al livello comando tra C, L, X
	SYSTEM.CHARSET	Necessario se il programma usa WCHAR o WSTRING (cfr. par. 2.6)
U(SER RESTART	Stessi files del comando da ripetere	Stesse posizioni del comando da ripetere
INITIALIZE	SYSTEM.PASCAL	seconda tappa
	SYSTEM.APPLE	prima tappa
	SYSTEM.MISCINFO	seconda tappa

Il disco di caricamento (BOOT) *si trova nel drive* 1 *dell'unità di controllo dello slot* 6 (cfr. par. 2.2).

• Contenuto dei dischi di sistema forniti con il Pascal

I files corrispondenti ai comandi sono disposti sui dischi APPLE come segue:

```
APPLE0:
SYSTEM.PASCAL     41 22-Sep-80    6 Code
SYSTEM.MISCINFO    1  4-May-79   47 Data
SYSTEM.COMPILER   75 19-Sep-80   48 Code
SYSTEM.EDITOR     47 24-Sep-80  123 Code
SYSTEM.FILER      28 18-Sep-80  170 Code
```

```
SYSTEM.LIBRARY     34 19-Sep-80  198 Data
SYSTEM.CHARSET      2 14-Jun-79  232 Data
SYSTEM.SYNTAX      14  1-Aug-80  234 Data
< UNUSED >         32            248
8/8 files, 32 unused, 32 in largest

APPLE1:
SYSTEM.APPLE       32  9-Nov-80    6 Data
SYSTEM.PASCAL      41 22-Sep-80   38 Code
SYSTEM.MISCINFO     1  4-May-79   79 Data
SYSTEM.EDITOR      47 24-Sep-80   80 Code
SYSTEM.FILER       28 18-Sep-80  127 Code
SYSTEM.LIBRARY     34 19-Sep-80  155 Data
SYSTEM.CHARSET      2 14-Jun-79  189 Data
SYSTEM.SYNTAX      14  1-Aug-80  191 Data
< UNUSED >         75            205
8/8 files, 75 unused, 75 in largest

APPLE2:
SYSTEM.COMPILER    75 19-Sep-80    6 Code
SYSTEM.LINKER      24 16-Sep-80   81 Code
SYSTEM.ASSMBLER    54 19-Sep-80  105 Code
6500.OPCODES        2 20-Dec-78  159 Data
6500.ERRORS         7 28-Mar-79  161 Data
< UNUSED >        112            168
5/5 files, 112 unused, 112 in largest

APPLE3:
SYSTEM.APPLE       32  9-Nov-80    6 Data
FORMATTER.CODE      4 14-Aug-80   38 Code
FORMATTER.DATA      6 14-Aug-80   42 Data
LIBRARY.CODE        8 15-Sep-80   48 Code
LIBMAP.CODE         9  1-Aug-80   56 Code
SETUP.CODE         33  7-Feb-79   65 Code
BINDER.CODE         5  1-Aug-80   98 Code
CALC.CODE           8 28-Dec-78  103 Code
LINEFEED.TEXT       4  1-Aug-80  111 Text
LINEFEED.CODE       2  1-Aug-80  115 Code
SOROCGOTO.TEXT      4 29-Mar-79  117 Text
SOROCGOTO.CODE      2 14-Aug-80  121 Code
SOROC.MISCINFO      1 13-Mar-79  123 Data
HAZELGOTO.TEXT      4 29-Mar-79  124 Text
HAZELGOTO.CODE      2 14-Aug-80  128 Code
HAZEL.MISCINFO      1 19-Mar-79  130 Data
CROSSREF.TEXT       8  4-Mar-80  131 Text
CROSSREF.CODE       3  1-Aug-80  139 Code
SPIRODEMO.TEXT      6  4-May-79  142 Text
```

```
SPIRODEMO.CODE       2 14-Aug-80  148 Code
HILBERT.TEXT         6  4-May-79  150 Text
HILBERT.CODE         2 14-Aug-80  156 Code
GRAFDEMO.TEXT       28  4-May-79  158 Text
GRAFDEMO.CODE       12 14-Aug-80  186 Code
GRAFCHARS.CODE       3 14-Aug-80  198 Code
GRAFCHARS.TEXT       6 22-Jun-79  201 Text
TREE.TEXT            8 22-Jun-79  207 Text
TREE.CODE            3 14-Aug-80  215 Code
BALANCED.TEXT       12 22-Jun-79  218 Text
BALANCED.CODE        4 14-Aug-80  230 Code
DISKIO.TEXT         22 14-Aug-80  234 Text
DISKIO.CODE          7 14-Aug-80  256 Code
< UNUSED >          17            263
32/32 files, 17 unused, 17 in largest
```

Il disco APPLE 0	: contiene i files necessari per editare e eseguire un programma.
Il disco APPLE 1	: contiene i files necessari per caricare il sistema e editare un programma.
Il disco APPLE 2	: contiene i files necessari per compilare o assemblare un programma.
APPLE 0	: è utilizzato con un sistema monodrive.
APPLE 1 e APPLE 2	: sono utilizzati con un sistema multidrive (cfr. par. 2.2).
Il disco APPLE 3	: contiene dei programmi utility (formattamento dischi...).

2.3.4 Comandi utilizzabili a tutti i livelli

• Gestione dello schermo

Lo schermo standard dell'APPLE è di 40 caratteri per linea. Quello del sistema Pascal è di 80 caratteri per linea. Ci sono dunque due pagine di schermo. Potete passare da una pagina all'altra battendo CTRL-A. Si torna alla prima pagina battendo nuovamente CTRL-A.
Il comando CTRL-Z consente di avere una pagina continua sullo schermo. Lo spostamento si effettua seguendo il cursore. Questo comando viene annullato dal CTRL-A.

• Interruzione e rilancio di un programma

Interruzione di un programma: il comando CTRL-A provoca l'interruzione del programma in corso e la visualizzazione del messaggio:

PROGRAM INTERRUPTED BY USER

Dovete allora battere uno spazio e il sistema si reinizializza.

Interruzione dei messaggi e delle stampe: il comando CTRL-F blocca l'uscita dei messaggi e delle stampe del programma. Quest'ultimo non viene fermato e l'uscita dei risultati e dei dati riprende col prossimo CTRL-F.

Interruzione e ripresa di un programma: il comando CTRL-S consente di fermare momentaneamente l'esecuzione di un programma (per esempio per studiare i dati intermedi). Il programma riparte ribattendo CTRL-S.

2.4 IL SISTEMA DI GESTIONE DEI FILES

2.4.1 Generalità

Il sistema di gestione dei files è la parte del sistema Pascal che gestisce i dischi, le periferiche e il trasferimento di dati tra periferiche. Creare un file, distruggerlo, cambiare i nomi dei files, listare un file su una periferica è di competenza del FILER (sistema di gestione dei files).
Il sistema di gestione dei files viene richiamato battendo il comando F(ILER. Il file SYSTEM.FILER viene così caricato in memoria e potete togliere il disco che lo contiene.
Ottenete sullo schermo:

FILER: G,S,N,L,R,C,T,D,Q,

oppure

(FILER: G(ET, S(AVE, W(HAT, N(EW, L(DIR, R(EM, C(HNG,
T(RANS, D(ATE, Q(UIT)

Battete ? Compariranno altri comandi del "FILER"

FILER: W,B,E,K,M,P,V,X,Z

oppure

(FILER: B(LAD-BLDS, E(XIT-DIR, K(RNCH, M(AKE,
P(REFIX, V(OLS, X(AMINE, Z(ERO)

Vedremo il significato di questi comandi nel paragrafo 2.4.5.
L'unico comando che vi occorre attualmente è il comando Q(UIT che porta a lasciare il sistema in gestione di files e a ritornare al livello comando. Per questo motivo il file SYSTEM.PASCAL deve essere sul disco di caricamento nel drive 1 dell'unità di controllo dello slot 6 (cfr. par. 2.3).

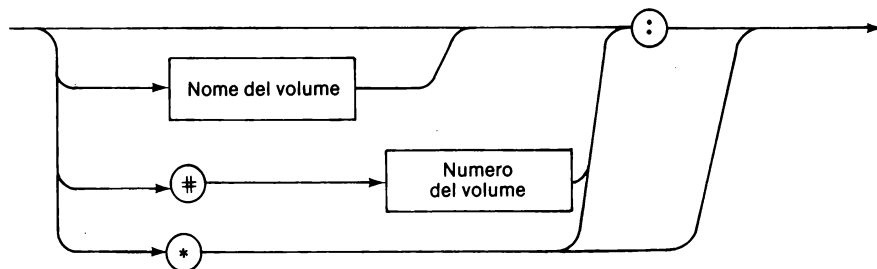
2.4.2 I volumi

Nella descrizione dei comandi del FILER chiameremo volume una periferica di Input/Output.
Per esempio, considereremo lo schermo, la tastiera o un disco come volumi. Un volume può essere referenziato dal suo nome o dal suo numero. Vi diamo qui sotto la relazione tra numero di volume e periferica.

Numero volume	Nome volume	Descrizione della periferica
0:		(Nome utilizzato)
1:	CONSOLE:	Schermo/tastiera con echo di tastiera
2:	SYSTEM:	Schermo/tastiera senza echo di tastiera
3:		(Nome utilizzato)
4:	<nome disco>:	Unità di controllo slot 6 drive 1
5:	<nome disco>:	Unità di controllo slot 6 drive 2
6:	PRINTER:	Stampante (scheda d'interfaccia slot 1)
7:	REMIN:	Entrata scheda di interfaccia
8:	REMOUT:	Uscita slot 2
9:	<nome disco>:	Unità di controllo slot 5 drive 1
10:	<nome disco>:	Unità di controllo slot 5 drive 2
11:	<nome disco>:	Unità di controllo slot 4 drive 1
12:	<nome disco>:	Unità di controllo slot 4 drive 2

Parecchi comandi del sistema di gestione dei files necessitano di almeno un volume. La specificazione di un volume consiste nel fornire un numero o un nome di volume seguito da :. I due punti (:) differenziano il nome di una periferica da quello di un file. Il fatto di non specificare il nome di un file dopo i due punti (:), indica che il comando deve essere applicato all'intero disco.
Il nome del volume deve essere composto da meno di 7 caratteri e non deve contenere i 4 caratteri seguenti:

$ = ? ,

La classificazione di un volume segue il seguente diagramma:

1. *Un asterisco * sostituisce il nome del disco di caricamento.*
2. *È possibile (comando* FILER PREFIX) *fissare il nome di un disco per default. Questo disco verrà richiamato battendo solo:* (cfr. par. 2.4.5).

2.4.3 I files

• I diversi tipi di files

Un file è un insieme di informazioni memorizzate su disco e designate dal

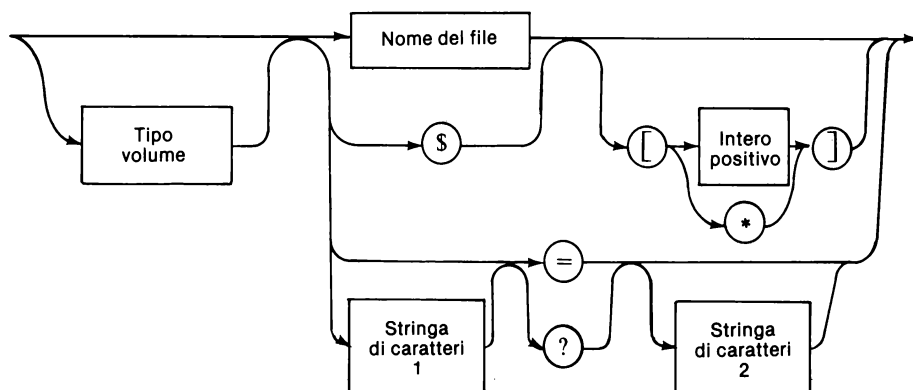
nome del file. Ogni disco contiene un catalogo in cui sono inseriti i nomi dei files e le informazioni necessarie alla loro gestione.
L'uso di un file è determinato dal suo tipo. I seguenti tipi sono accettati dal sistema Pascal.

Suffisso	Tipo del file	Nome del tipo
•TEXT	File sorgente editabile	TEXTFILE
•CODE	File in P CODE	CODEFILE
•DATA	Dati	DATAFILE
•BAD	Settori fisici danneggiati	BADFILE
•INFO	(non utilizzato)	INFOFILE
•GRAF	(non utilizzato)	GRAFFILE
•FOTO	(non utilizzato)	FOTOFILE

• Il file di lavoro "WORKFILE"

C'è un file che viene utilizzato unicamente come file di lavoro; è il SISTEM.WRK.TEXT (o .CODE). Questo file, se è presente sul disco, viene preso per default durante l'uso dei comandi R(UN, E(DIT, C(OMPILE, A(SSEMBLE e L(INK.
Il sistema di gestione dei files (FILER) può distruggere il file di lavoro, memorizzarlo e designare il nuovo file da caricare nel file di lavoro.

• Specificazione di un file

Numerosi comandi del sistema Pascal richiedono l'indicazione del nome di un file o la sua specificazione. Quest'ultima è costituita dal nome o dal numero di volume, seguito dal nome del file. Il diagramma seguente definisce l'esatta sintassi:

Esempio:

4 : nome file 5
APPLES: 5:S*
BLA = BLA
BLA?NC

Nome del file. Il nome di un file è costituito da una stringa di 15 caratteri al massimo. Gli ultimi cinque devono essere .TEXT (file sorgente) o .CODE (file in P-code).
Nel nome di un file, sono ammessi tutti i caratteri. Tuttavia, non dovete digitare nomi di files contenenti i caratteri ( ) parentesi, ($) simbolo del dollaro, (=) carattere uguale, (?) punto di domanda, (RETURN) e i caratteri di controllo CTRL-S, -M, -F, -C, -U, -A.
I caratteri sono interpretati in modo diverso dal sistema di gestione dei files Pascal dell'APPLE II. Esso non è in grado di trattare files il cui nome contiene uno dei seguenti caratteri $, = ?

Dimensione di un file. È possibile specificare la dimensione di un file mettendola tra parentesi.
Per esempio, la specificazione BLABLA [4] indica che il file BLABLA ha una dimensione di 4 blocchi di 5102 bytes (cfr. par. 2.4.5).
Esistono delle speciali possibilità riguardanti il numero dei blocchi. Se mettete BLABLA [0], il sistema di gestione dei files prende, come dimensione e spazio, quelli della massima zona libera sul disco. Questa è l'opzione presa per default.
Se indicate come numero di blocchi un asterisco (ad esempio BLABLA [*]), il sistema assegna al file la zona più grande scelta tra la metà della massima zona libera sul disco e la seconda massima zona libera sul disco.
La specificazione della dimensione di un file è utilizzata soprattutto per i comandi T(RANSFER e M(AKE (trasferimento e creazione del file) del FILER.

Specificazione di più files contemporaneamente. È possibile designare diversi files contemporaneamente usando il carattere uguale (=) e il punto di domanda (?) come "jolly". Questo è limitato ad un'occorrenza per nome di file.
Per esempio, se date come nome di file

BLA?BLA

oppure

BLA = BLA

verranno trattati tutti i file il cui nome comincia con BLA e finisce con BLA.
Se avete sul vostro disco i seguenti files:

BLABLA
BLBLA
BLASERBLA
LASERBAAST
BLABLA1

e digitate BLA=BLA, corrisponderanno alla vostra specificazione i files BLABLA e BLASERBLA.

2.4.4 Richiesta di informazioni su volumi e dischi

• Lista dei volumi

Se battete V(OLS quando il sistema di gestione dei files è in attesa di un comando (cfr. par. 2.4.1 *Generalità*), il sistema visualizzerà sullo schermo i volumi attivi (cfr. par. 2.4.2 *Volumi*).
Come un APPLE II monodrive, vedrete comparire sullo schermo:

```
VOLS       ON-LINE
 1         CONSOLE:
 2         SYSTERM:
 4         APPLE0:
ROOT VOL   IS-APPLE0:
PREFIX     IS-APPLE0:
```

Il volume chiamato "ROOT VOL" è il volume del disco di caricamento. Quello chiamato "PREFIX" è il volume corrente attivo del disco (cfr. par. 2.4.9).
Con un APPLE II e diversi mini floppy, otterrete per esempio sullo schermo:

```
VOLS       ON-LINE
 1         CONSOLE:
 2         SYSTERM:
 4         #APPLE1
 5         #PROJETS
 6         PRINTER
```

• Catalogo di un disco

Il comando L(IST del sistema di gestione dei files consente di avere il catalogo di un disco, completo o in parte. Quando battete L al livello comando del FILER (cfr. par. 2.4.1), il sistema chiede

DIR LISTING OF? (oppure DIR LISTING OF WHAT VOL?)

Dovete allora specificare un volume e, se lo desiderate, delle informazioni restrittive.
Supponiamo che il disco APPLE0: sia nel drive utilizzato per il caricamento del sistema (#4). Quando vedete il messaggio:

DIR LISTING OF?

potete rispondere indifferentemente *, #4, APPLE0:. Comparirà l'intero catalogo dell'APPLE0: nel seguente formato:

Nome del file	Numero di blocchi da 5120 utilizzati	Data dell'ultima modifica
APPLE0:		
SYSTEM.PASCAL	36	4-MAY-79
SISTEM.MISCINF	1	4-MAY-79
SYSTEM.COMPILER	71	30-MAY-79
SYSTEM.EDITOR	45	29-JAN-79
SYSTEM.FILER	28	24-MAY-79
SYSTEM.LIBRARY	36	22-JUN-79
SYSTEM.CHARSET	2	14-JUN-79
SYSTEM.SYNTAX	14	18-APR-79
SYSTEM.WRK.TEXT	10	20-FEB-82

9/9 FILES, 31 blocks UNUSED, 23 blocks IN LARGEST

Avete anche la possibilità di chiedere solo una parte del catalogo; se, ad esempio, rispondete:

DIR LISTING OF? SYSTEM.C=

otterrete sullo schermo il seguente risultato

```
APPLE0:LINE
SYSTEM.COMPILER     71     30-MAY-79
SYSTEM.CARSET        2     14-JUN-79
2/9 FILES, 45 UNUSED, 45 IN LARGEST
```

Per quel che riguarda lo spazio disponibile su disco, le cifre indicate dal sistema durante la visualizzazione del catalogo parziale, sono falsate in quanto l'ultimo file visualizzato viene preso come ultimo file del disco e la parte restante del disco viene considerata libera.

Per ottenere lo spazio disponibile su un disco, occorre quindi, richiederne il catalogo completo.

È possibile far uscire il catalogo su un altro volume, come ad esempio un file o una stampante.

Se usate il comando

DIR LISTING OF? #4:S=R,PRINTER:

otterrete il messaggio:

```
APPLE0:
SYSTEM.COMPILER     71     30-MAY-79
SYSTEM.EDITOR       45     29-JAN-79
SYSTEM.FILER        28     24-MAY-79
3/11 FILES, 93 UNUSED, 93 IN LARGEST
```

Se utilizzate il comando

DIR LISTING OF? #4:, #4:DIR.TEXT

viene copiato nel file DIR.TEXT il catalogo del volume #4 e compare il seguente messaggio:

WRITING...

• **Catalogo esteso di un disco**

Il comando E del sistema di gestione dei files della scheda Pascal (cfr. par. 2.4.1) consente di ottenere un catalogo più completo di quello fornito dal comando L (cfr. par. 2.4.4).
La sintassi degli scambi con il sistema per il comando E, è uguale a quella del comando L.

Esempio:

DIR LISTING OF? #4:

Nome file	Nome dei blocchi da 256 bytes presi	Data	N° 1 blocco	Tipo
APPLE0:				
SYSTEM.PASCAL	36	4-MAY-79	6	DATA
SYSTEM.MISCINFO	1	4-MAY-79	42	DATA
SYSTEM.COMPILER	71	30-MAY-79	43	CODE
SYSTEM.EDITOR	45	29-JAN-79	114	CODE
SYSTEM.FILER	28	24-MAY-79	159	CODE
SYSTEM.LIBRARY	36	22-JUN-79	187	DATA
SYSTEM.CHARSET	2	14-JUN-79	223	DATA
SYSTEM.SYNTAX	14	18-APR-79	225	TEXT
<UNUSED>	12		239	
SYSTEM.WRK.TEXT	10	2-JUL-81	251	TEXT
<UNUSED>	19		261	

9/9 FILES, 27 UNUSED, 19 IN LARGEST

Se avete uno schermo da 80 caratteri (scheda di espansione), vi viene dato in più il numero di bytes occupati nell'ultimo blocco di ogni file.

2.4.5 Struttura delle informazioni su un disco gestito dal FILER

In questo paragrafo tratteremo l'organizzazione dei dati su un disco e alcuni punti lasciati in sospeso nei paragrafi precedenti.

Un disco è suddiviso in 35 tracce da 16 settori, ognuno dei quali è costituito da 256 bytes di dati. Il disco contiene 140 K bytes di dati utili. Ogni settore contiene una sezione indirizzo, costituita dal numero della pista e da quello del settore nella pista, e i dati.
Il sistema Pascal utilizza dei blocchi di due settori, cioè 512 bytes (1/2 K bytes). Vi si può accedere usando le funzioni UNITWRITE e UNITREAD.
I 280 blocchi di un disco non sono tutti utilizzabili per la memorizzazione dei vostri files.
I blocchi 0 e 1 contengono il programma d'inizializzazione dell'APPLE II che carica in memoria il sistema Pascal.
I blocchi da 2 a 5 (2 K bytes = 2 blocchi) contengono il catalogo del disco e tutte le informazioni necessarie alla gestione dei vostri files. È possibile gestire fino a 77 files per disco.
Per semplificare il sistema di gestione dei files FILER, i vostri files vengono memorizzati su blocchi contigui di piste vicine.
Questa scelta ha, per l'utente, un'importante conseguenza. Per creare, modificare, ecc. dei files, dovete sempre verificare le zone libere sul disco. Un disco è un alternarsi continuo di files e "buchi" catalogati tramite il comando E (cfr. par. 2.4.4). Il fatto di avere numerosi "buchi" non permette di memorizzare un file di grosse dimensioni.
Quando avete un file da modificare con l'editor, dovete innanzitutto verificare che esista sul disco una zona libera superiore alla dimensione del vostro file. Cosa accade quando abbandonate l'editor con uno dei comandi W(RITE o U(DATE? Il sistema di gestione dei files memorizza la nuova versione del file e distrugge la precedente. Il disco deve poter contenere le due versioni; altrimenti, non potrete memorizzare le vostre modifiche. Vi consigliamo dunque, di conservare sempre una zona libera circa uguale al doppio della dimensione del file da editare.
Esiste un comando del sistema di gestione dei files che consente di spostare i files e di collegare le zone libere per crearne una più grande.

2.4.6 Comandi generali del sistema di gestione dei files

• Raggruppamento delle zone libere K(RUNCH

Questo comando, spostando i files necessari, raggruppa le diverse zone libere alla fine del disco. Consente in questo modo, di creare delle zone libere maggiormente utilizzabili.
Il comando K necessita solamente del nome di un volume di disco o di un numero di volume e visualizza i nomi dei files spostati sul disco.

1. *Prima di digitare questo comando, per non rischiare di danneggiare un file, è utile verificare l'esistenza di blocchi rovinati e fissarli* (cfr. par. 2.4.8).
2. *Non aprire mai lo sportello del drive, premere* RESET, *ecc., prima che il comando* K *sia terminato. Il vostro disco rischierebbe di essere distrutto definitivamente. Se il file* SYSTEM.PASCAL *è stato spostato, reinizializzate il sistema.*

3. *È possibile creare la zona libera attorno ad un blocco diverso dal blocco* 280, *fornendo il numero del blocco; i files verranno raggruppati all'inizio e alla fine del disco per lasciare spazio attorno al vostro blocco.*

Esempio:

K
CRUNCH ? *
oppure #4:
FROM END OF DISK, BLOCK 280? (Y/N)

Y *lancia l'operazione normale alla fine del disco,*
N *visualizza il messaggio*

STARTING AT BLOCK?

dovrete allora digitare il numero del blocco.

• Creazione di un file

Il comando M(ake riserva un'entrata nel catalogo e una zona sul disco sotto un dato nome di file. Questo è un comando utilizzato più che altro per riservare dello spazio su un disco, in modo da poterlo usare in futuro oppure recuperare gli errori di distruzione di files.

Esempio:

M
MAKE WHAT FILE? #4:PLACE.TEXT [32]

Il file PLACE.TEXT di 32 blocchi viene creato nel primo posto libero maggiore di 32 blocchi.
Ad ogni tipo di file viene associato un formato; consultateli per sapere quanto spazio dovete riservare (cfr. par. 2.4.10).

• Cambio del nome di un file, di un disco

Il comando C(HANGE consente di cambiare il nome di un file o il nome di un volume di disco. Esso necessita l'indicazione di due specificazioni di files oppure di due nomi di dischi. Le due specificazioni devono essere separate da una virgola o da un Return. Il nome da cambiare deve comparire al primo posto. Se si tratta della specificazione di un file, ogni numero o nome di volume dato nella seconda specificazione vengono ignorati (devono essere uguali).
Usando i caratteri jolly = e ? potete modificare il nome di più files con un solo comando.
Il comando C(HANGE non viene eseguito se il nuovo nome supera 15 caratteri. Viene visualizzato il messaggio NOT PROCESSED.

Esempi:

1. CHANGE? APPLE5:, APPLE4:

se il disco APPLE5: si trova in un drive, otterrete la seguente risposta

APPLE5:→APPLE4;

Il disco APPLE5: è stato ridenominato APPLE4:

2. CHANGE? APPLE0:EXEMPLE, APPLE1:EXEMPLE1

Il file APPLE0:EXEMPLE prenderà il nome APPLE0:EXEMPLE1 e non APPLE1:EXEMPLE1. Il secondo volume viene ignorato.

3. Supponiamo abbiate il seguente disco:

LIBRO:
CH1.TEXT
CH3.TEXT
CH7.TEXT
INTRO.TEXT

Il comando

CHANGE? CH = TEXT,CH = OLD.TEXT

creerà il seguente catalogo

LIBRO:
CH1.OLD.TEXT
CH3.OLD.TEXT
CH7.OLD.TEXT
INTRO.TEXT

Se a questo punto digitate il comando

CHANGE? CH =
TO WHAT? CHAPITRE =

La modifica non verrà presa in considerazione e otterrete il messaggio

BAD DEST FOR FILES FOUND

I simboli (=), (?) rappresentano per il comando C la stessa stringa di caratteri alla partenza e all'arrivo (copia di una parte del nome dei files).
Se si cambia il nome del disco di caricamento o del disco corrente (cfr. Comando Prefix), *il comando* C(hange *modifica automaticamente questi dati.*

• Distruzione di un file

Il comando R(emove consente di distruggere uno o più files di un disco restituendo lo spazio occupato dai files, che viene così considerato come libero, e liberando gli spazi nel catalogo.
Il comando R(emove necessita di una specificazione di files che può includere i caratteri jolly = e ?. Le informazioni relative alla dimensione del file sono ignorate.

Esempio: supponiamo di voler distruggere il file creato nel paragrafo 2.2:

APPLE0:ARAROM.TEXT

Premiamo R
Il filer risponde

REMOVE? (o REMOVE WHAT FILE?)

digitiamo allora APPLE0:ARAROM =
Il FILER cerca il (i) file(s) e, a titolo di verifica, vi pone la seguente domanda

UPDATE DIRECTORY?

Se rispondete Y, i files vengono distrutti; altrimenti il comando viene ignorato.
Non utilizzare mai il comando Remove *per cancellare il file di lavoro. Per sostituire il file di lavoro usate i comandi speciali* New *e* Get. *Altrimenti andrete incontro a un disastro.*

- **Rimessa a zero di un disco**

Il comando Z(ero scrive degli zero sul catalogo del disco. Il suo contenuto viene perso e tutti i blocchi vengono resi liberi. Questo comando però, non formatta il disco. Esso è equivalente al seguente comando:

REMOVE? =

Esempio:

Z
ZERO DIR OF? (o ZERO DIR OF WHAT VOL?)
4:
DESTROY APPLE5:? Y (in altro caso abbandona il comando)
DUPLICATE DIR? N (il sistema APPLE non l'accetta)
ARE THERE 280 BLKS ON THE DISK? (Y/N) Y

Per quel che riguarda questa domanda, se il numero è diverso da 280, conviene reinizializzare il disco.

NEW VOL NAME? APPLE4:
APPLE4:CORRECT? Y (altrimenti riprende il processo)
APPLE4: ZEROED

2.4.7 I comandi di manipolazione del "Workfile"

- **Rimessa a zero del file di lavoro**

Il comando N(ew del FILER cancella il file di lavoro in modo tale che nessun file possa essere trovato dal text editor, dal compilatore Pascal o dall'assemblatore. Il precedente contenuto del file di lavoro è perso e non esiste più il file SYSTEM.WRK = fino alla prossima creazione del file di lavoro (Text editor o comando G(et).

Se battere N(ew quando esiste un file di lavoro non vuoto sul disco di caricamento, il sistema vi domanda

THROW AWAY CURRENT WORKFILE?

Se rispondete Yes, il (i) file(s) di lavoro del disco di caricamento vengono distrutti.

• Stato del file di lavoro

Il comando W(hat consente di ottenere l'identificazione (nome file) e lo stato (memorizzato/o no) del file di lavoro.
Se il file di lavoro è stato memorizzato su un disco diverso da quello di caricamento, il suo stato sarà "non memorizzato" in quanto continua a esistere sul disco di caricamento.

• Sostituzione del file di lavoro

Il comando G(et del sistema di gestione dei files consente di identificare un file come file di lavoro. Questo file dovrà essere presente su un volume, alla prossima utilizzazione del text editor, del compilatore o alla prossima esecuzione. Se in quel momento il file non è presente, compare il seguente messaggio:

ERROR: WORKFILE LOST

Questo file è necessario alla sua prima utilizzazione come file e non al momento del G(et.
Se nel momento in cui fate un G(et, *esiste già un file di lavoro* SYSTEM.WRK *sul disco di caricamento, compare il seguente messaggio*

THROW AWAY CURRENT WORKFILE? (cfr. N(ew)

Non è necessario precisare un tipo (.TEXT o .CODE). Per il file di lavoro saranno tutti caricati e otterrete uno dei seguenti messaggi

	TEXT	FILE LOADED
o	TEXT CODE	FILE LOADED
o	CODE	FILE LOADED

• Memorizzazione del file di lavoro

Il comando S(ave memorizza tutte le versioni del file di lavoro del disco di caricamento sotto il nome specificato col G(et (cfr. par. 2.4.7) o sotto un nuovo nome.
Se sotto il nome precisato esiste già un file, il sistema vi chiede conferma. In questo caso, la versione precedente viene soppressa e viene creata la nuova versione del file.
Se memorizzate il file di lavoro sul vostro disco di caricamento, il o i files che compongono il file di lavoro vengono nominati. Altrimenti, c'è l'effettivo trasferimento e il file di lavoro resta immutato.

Non specificare mai il tipo (.TEXT *o* .CODE) *del file su cui si memorizza il file di lavoro. Vengono aggiunti automaticamente per ogni tipo.*

2.4.8 Gestione della validità fisica dei dischi

La gestione della validità fisica dei dischi viene effettuata blocco per blocco. I blocchi danneggiati vengono messi in un file il cui nome ha il suffisso .BAD e non è più possibile spostarli (fissaggio dei blocchi danneggiati).

- **Analisi di una zona fisica del disco**

Il comando B(ad blocks cerca i blocchi difettosi di un volume confrontando il codice somma del blocco (CRC o uno equivalente) con quello che verrà calcolato per il blocco. Questo comando richiede un nome o un numero di volume.

Esempio:

```
B
BAD BLOCK SCAN OF? APPLE0:          (nome del disco)
SCAN FOR 280 BLOCKS? (Y/N) Y        (tutto il disco?)
```

Se il numero di blocchi specificato è diverso da 280, probabilmente il vostro disco è danneggiato. Se rispondete Y, la ricerca dei blocchi rovinati viene eseguita sull'intero disco; altrimenti, il sistema vi domanda qual è l'intervallo da analizzare.
Nel 90% dei casi, non otterrete errori e apparirà il seguente messaggio

```
0 BAD BLOCKS
```

Se un blocco è danneggiato, sentirete schioccare il drive e vedrete comparire un messaggio simile al seguente:

```
BLOCK 45 IS BAD
BLOCK 140 IS BAD
2 BAD BLOCKS
FILE(S) ENDANGERED:
SYSTEM.COMPILER      43    113
SYSTEM.EDITOR       114    158
```

Ottenete cioè, la lista dei blocchi danneggiati e quella dei files in pericolo. Niente è però definitivo: i blocchi danneggiati non sono contrassegnati e potete tentare di recuperare i vostri files. Comunque, se avevate già salvato questi files su un altro disco, non tentate di recuperare i blocchi rovinati e contrassegnateli utilizzando il comando X(amine (cfr. prossima sezione).

- **Fissaggio dei blocchi danneggiati di un disco**

Il comando X(amine fissa i blocchi rovinati di un disco. Questo comando richiede il nome o il numero di volume di un disco.

Esempio:

```
X
EXAMINE BLOCKS ON? APPLE0:
(o EXAMINE BLOCKS ON WHAT VOLUME?)
BLOCK-RANGE?
```

A questo punto avrete già usato il comando B(ad blocks e quindi sarete a conoscenza dei blocchi danneggiati; non dovrete far altro che indicarli come risposta.
Nel nostro esempio risponderemo 45-140.
Tutti i blocchi compresi nell'intervallo saranno letti sul disco, riscritti allo stesso posto e riletti una seconda volta; se paragonando le due letture viene trovato un errore, il blocco viene contrassegnato e considerato come un blocco rovinato.
Con il comando eXamine, il comando B(ad Blocks *risulta superfluo. Tuttavia a volte si rivela utile in quanto è molto più rapido e può rilevare altri errori.*
Alla fine del comando X, *i blocchi esaminati vengono diagnosticati:*

```
BLOCK 45 IS BAD
BLOCK 46 MAY BE OK
          .
          .
          .
BLOCK 139 MAY BE OK
BLOCK 140 IS BAD
FILE(S) ENDANGERED:
SYSTEM.COMPILER      43  113
SYSTEM.EDITOR       114  158
MARK BAD BLOCKS? (FILES WILL BE REMOVED!)
(Y/N) Y
```

Il sistema vi offre la possibilità di togliere i files in pericolo dal catalogo e isolare i blocchi danneggiati nei files non trasferibili. Nel nostro esempio, vengono distrutti i files SYSTEM.COMPILER e SYSTEM.EDITOR e creati i files BAD.00045.BAD e BAD.00140.BAD.
Un blocco dichiarato MAY BE OK *segnala che (probabilmente) non ci sono errori fisici sui settori del blocco; potrebbero, però, esserci degli errori logici. Se il campo indirizzo di un settore di un disco diventa illeggibile, il blocco viene rivelato dal comando* B(ad Blocks, *ma non può essere né fissato né raggiunto. L'unica soluzione è quella di reinizializzare il disco.*

2.4.9 Altri comandi

• Designazione di un disco corrente

Il comando P(refix consente di designare un disco come disco corrente; sarà, cioè, il disco su cui si andranno a cercare i files dei quali non è stato

specificato il volume. Questo comando richiede l'indicazione di un nome o un numero di volume. Non è necessario che quest'ultimo sia attivo in quell'istante.
Per utilizzare il disco corrente, è sufficiente indicare il simbolo (:).
Il disco corrente viene inizializzato col nome del disco di caricamento. Se cambiate il nome al disco corrente con il comando C(hange, automaticamente viene lanciato il comando P.

• Uso e aggiornamento della data

Il comando D(ate permette di modificare la data utilizzata dal sistema. È possibile fissare il giorno, il mese e/o l'anno.

Esempio:

```
DATE SET:<1..31>-<JAN..DEC>-<00..99>
TODAY IS 25-JUN-81
NEW DATE? 26          (cambia il giorno)
          -JUL        (cambia il mese)
          --82        (cambia l'anno)
          1/JAN/83    (cambia i 3 parametri)
```

Il trattino e lo slash (/) possono servire da separatori. Potete cambiare la parte di data che v'interessa. Dopo aver battuto Return, compare sullo schermo la nuova data.

2.4.10 Formato dei files sui dischi

• Files di tipo TEXT

All'inizio di un file di tipo TEXT, si trovano due blocchi il cui uso è interno al text editor e che sono rispettati da tutte le parti del sistema.
Quando un programma utente apre un file il cui nome termina per .TEXT e si posiziona all'inizio mediante REWRITE e RESET, il sistema sorpassa questa zona per posizionarsi subito dietro.
Inoltre quest'intestazione non viene trasmessa durante una visualizzazione sullo schermo o una scrittura su stampante, ma al momento del trasferimento da disco a disco.
Dopo l'intestazione si trova il testo vero e proprio, suddiviso in pagine di 1024 bytes secondo il seguente formato:

DLE margine testo CR DLE margine testo CR ... spazi

Gli spazi a fine pagina sono presenti per rendere intero il numero di linee per pagina (non devono esserci linee a cavallo di due pagine da 1 K byte imposto dal compilatore).
DLE margine consente di precisare una larghezza di margine. Non serve se non c'è un margine fissato; altrimenti si mette come margine 32, più la larghezza reale del margine.

• **Files di tipo DATA**

I formati dei files di tipo DATA sono a libero arbitrio dell'utente.

• **Files di tipo CODE**

Un file di tipo CODE è costituito da segmenti; il loro numero può arrivare fino a 16. Il blocco zero di un file di tipo CODE contiene le informazioni necessarie alla gestione dei 16 segmenti di codice. Queste informazioni sono rappresentate come segue:

```
RECORD
DISKINGO  : ARRAY 0..15 OF RECORD
            CODELENG,     (*LUNGHEZZA DEL SEGMENTO*)
            CODEADDR      (*INDIRIZZO DEL SEGMENTO*)
            : INTEGER END;
SEGNAME   ; ARRAY 0..15 OF PACKED ARRAY 0..7 OF CHAR;
            (*TABELLA DEI NOMI DEL PROGRAMMA PRINCI-
            PALE E DELLE PROCEDURE COMPILATE*)
SEGKIND   : ARRAY 0..15 OF (LINKED, HOSTSEG, SEGPROC,
            UNITSEG, SEPRTSEG, UNLINKED-INTRINS, LIN-
            KED-INTRINS, DATASEG);
            (*TABELLA DEI TIPI DI SEGMENTI:
            LINKED              segmento interamente eseguibile
            HOSTSEG             programma ospite
            SEGPROC             una procedura PASCAL in un
                                segmento (non usato)
            UNITSEG             unità compilata regolarmente
            SEPRTSEG            unità compilata separatamente
            UNLINKED-INTRINS    unità contenente delle chiamate
                                a funzioni o a procedure esterne
            LINKED-INTRINS      unità intrinseca completa
            DATASEG             segmento di dati
TEXTADDR  : ARRAY 0..15 OF INTEGER;
            (*per un'unità, valore del numero di blocco dove comincia
            l'interfaccia con l'unità, altrimenti 0*)
SEGINFO   : PACKED ARRAY 0..15 OF PACKED RECORD
            SEGNUM: 0..255; (*Bits da 0 a 7 numeri di segmento per il
            numero di segmento di codice*)
            MTYPE:   0..15; (*TIPO DI P-CODE
                             0 vecchio
                             1 P code msb come primo (riempimento
                                    →)
                                    msb come secondo 0 nella parola
                                    (riempimento ←)
            UNUSED:  0..1;
            VERSION: 0..7    (*numero di versione*)
            END;
```

```
INTRINS-SEGS: SET OF 0..31; (*UNITÀ INTRINSECHE DELLA BI-
                                BLIOTECA NECESSARIE*)
COMMENT: STRING
END;
```

2.5 IL TEXT EDITOR

2.5.1 Generalità

Il text editor è piuttosto potente. Abbiamo già visto, nel paragrafo 1.2, un esempio di utilizzazione dell'editor. Riprenderemo ora, i vari comandi e spiegheremo le loro possibilità. Innanzitutto vi rammentiamo come far funzionare il text editor.

• Messa in funzione su un sistema monodrive

In generale, il procedimento da seguire per editare un file che non si trova sul disco di caricamento, è il seguente:

- cancellare il file di lavoro (comando N(ew del FILER);
- trasferire il file da editare sul disco di caricamento (comando T(ransfer del Filer);
- prendere questo file come file di lavoro (comando G(et del Filer);
- editare il file e fare le modifiche;
- lasciare l'editor aggiornando il file di lavoro (comandi Quit e Update dell'editor);
- memorizzare il file di lavoro (comando S(ave del Filer);
— trasferire in senso inverso i files corretti.

• Messa in funzione su un sistema multidrive

Il modo di procedere è lo stesso tranne per il fatto che il trasferimento viene effettuato direttamente dai comandi G(et e S(ave del Filer (cfr. par. 2.4.8). In questo modo, le manipolazioni di dischi sono di gran lunga alleggerite.

• Comandi dell'editor

Se siete al livello comando del sistema, per entrare nell'editor, dovete battere E(DIT. L'editor viene caricato; guarda se esiste un file di lavoro: se esiste, lo carica in memoria, visualizza la sua prima pagina di testo e si mette in attesa di un comando, altrimenti fa apparire il seguente messaggio:

```
>EDIT
NO WORKFILE IS PRESENT. FILE? (<RET> FOR NO
FILE <ESC-RET> TO EXIT)
```

Potete, a questo punto, introdurre il nome di un file. Il comando G(et è eseguito implicitamente. Se non mettete alcun nome, viene creato un nuovo file. La linea dei comandi dell'editor è la seguente:

>EDIT:A(DJST C(PY D(ELETE F(IND I(NSRT J(MP
R(PLACE Q(UIT X(CHNG Z(AP

In questo paragrafo, studieremo tutti i comandi.

• Gestione dello schermo e spostamento del cursore

Quando lavorate con l'editor, vedete comparire davanti a voi una pagina di schermo da 40 caratteri. Il text editor lavora con linee che possono arrivare fino a 80 caratteri. Utilizzeremo, per tutto il sistema, i caratteri CTRL-A e CTRL-Z per seguire il cursore sullo schermo.
Ad un dato istante, sullo schermo compare una pagina di testo; per poter accedere ad una frase, una parola o un carattere che si trova in mezzo allo schermo, per fare un inserimento, una cancellazione o una sostituzione di caratteri, potete muovere il cursore con i seguenti 4 caratteri:

CTRL-O : il cursore sale di una linea,
CTRL-L : il cursore scende di una linea,
→ (HT) : il cursore si sposta verso destra,
← (BS) : il cursore si sposta verso sinistra.

Battere un numero prima di questi caratteri porta alla ripetizione del carattere voluto. Battero uno slash (/) conduce ad una ripetizione infinita; in questo modo potete vedere tutti i testi che dovete modificare.
Per alcuni caratteri, viene presa per default una direzione e un verso di spostamento (in avanti o indietro). Al momento dell'entrata nell'editor, la direzione presa per default è in avanti.
Però, se battete uno dei caratteri , < - la direzione cambia verso sinistra, se battete . > + la direzione cambia verso destra.
Alcuni comandi o caratteri sono influenzati da questa direzione; sono i seguenti:

spazio avanza o indietreggia di uno spazio secondo la direzione,
CTRL-I si posiziona sulla tabulazione più vicina secondo la direzione,
Tab prima o dopo della posizione attuale del cursore,
Return sale o scende di una linea secondo la direzione.

Se battete il carattere uguale (=), posizionate il cursore sull'ultimo testo aggiunto, sostituito o cercato.

2.5.2 Comandi per la modifica del testo

• Inserimento di un testo

Generalità. Per inserire del testo in un file, dovete procedere in due tappe:

- posizionamento del cursore sul luogo desiderato per l'inserimento;
- premere I(nsert al livello comando dell'editor.

Vedrete comparire la seguente linea:

>INSERT:TEXT <BS> A CHAR, <DEL> A LINE
<ETX> ACCEPTS, <ESC> ESCAPES

Il testo digitato verrà, allora, inserito tra il carattere che si trova subito a sinistra del cursore e il carattere sul quale è situato il cursore.
Per procedere più rapidamente, il text editor non aggiorna immediatamente il testo sullo schermo ma vi crea un buco per la vostra inserzione. Il testo riapparirà nella sua nuova versione quando, per accettare l'inserimento, digiterete CTRL-C (<ETX>), oppure nella sua versione precedente se premerete ESC (annulla l'inserimento).
Se, durante l'inserimento del vostro testo, fate un errore, potete scegliere diverse soluzioni per eliminarlo:

- battere < - (<BS>), che annulla l'ultimo carattere;
- battere CTRL-X (<DEL>), che annulla l'ultima linea;
- battere ESC, che annulla il testo inserito, e I (per inserirne uno nuovo);
- battere CTRL-C, che accetta il testo, riposizionarsi sull'errore e correggerlo.

Il vostro inserimento, dopo essere stato accettato con CTRL-C, è disponibile nel buffer usato per copiare i testi. Avete dunque la possibilità di riprodurre in vari punti del file, il testo inserito, usando il comando C(OPY dell'editor. Questa possibilità non esiste se terminate con <ESC>, cioè annullando l'inserimento.
La dimensione massima di un file è di 38 blocchi cioè circa 18400 caratteri. Se vi avvicinate alla dimensione massima del file, vedrete comparire sullo schermo il seguente messaggio:

ERROR: PLEASE FINISH UP THE INSERTION. PLEASE PRESS <SPACEBAR> TO CONTINUE

A questo punto, per poter continuare l'inserimento, dovete battere uno spazio. Prima di uscire dal modo d'inserimento potete finire la vostra frase. Se dovete ancora aggiungere un testo molto lungo, dovete dividere il vostro file in due o più parti.
Se invece continuate l'inserimento del testo dopo il suddetto messaggio, il vostro buffer si satura e ricevete un messaggio più grave:

ERROR: BUFFER OVERFLOW!!!! PLEASE PRESS
<SPACEBAR> TO CONTINUE

A questo punto, dovete premere lo spaziatore per uscire dal modo di inserimento e qualunque tentativo di ritornarvi porterà al seguente messaggio d'errore:

ERROR: NO ROOM TO INSERT. PLEASE PRESS
<SPACEBAR> TO CONTINUE

Caratteri speciali. I caratteri ([) e (]) vengono utilizzati per le tabelle in Pascal in quanto esse, a prima vista, non sono accessibili da tastiera.

Sull'APPLE II+, per ottenerli dovete digitare CTRL-K per [e SHIFT-M per].

Formattamento dei testi. Lo schema di formattamento viene fissato usando i comandi S(ET e E(nvironment dell'editor (cfr. qui sotto). Ci sono da fare due scelte:

- Aggiustamento automatico del margine sinistro di una linea sotto il primo carattere diverso dallo spazio della linea precedente. Chiameremo questo schema aggiustamento automatico di margine.
- Uso dei margini e limitazione del testo tra i margini.

Queste scelte sono due parametri indipendenti, ma esamineremo ora i quattro casi possibili:

*AGGIUSTAMENTO AUTOMATICO SENZA L'USO DEI MARGINI

È l'opzione presa per default dall'editor ed è utile per scrivere dei programmi Pascal e Fortran.
Dovete terminare ogni linea battendo Return. Il cursore si posiziona sotto il primo carattere diverso dal blank della linea precedente. Ogni linea inizia allo stesso margine di quella sopra.
Per cambiare margine, è sufficiente battere uno o più spazi se volete aumentarlo, una o più frecce verso sinistra se volete diminuirlo oppure CTRL-Q se volete annullarlo.
Quando cominciate un inserimento, il margine considerato è quello della linea dove inizia l'inserimento.

Esempio: digitando i seguenti caratteri:

ONE <return> <space> <space> DUE <return> QUATTRO <return> < ← > CINQUE <return>

ottenete sullo schermo il seguente risultato

```
ONE
        DUE
        QUATTRO
        CINQUE
```

*USO DEI MARGINI SENZA AGGIUSTAMENTO AUTOMATICO DI UNA LINEA SULLA PRECEDENTE

È il metodo usato per scrivere testi quali lettere o altri documenti. Il testo viene inquadrato da un margine di paragrafo, un margine sinistro e un margine destro.
Se una parola supera il margine destro, viene inserito automaticamente il carattere RETURN prima della parola e di ciò che la segue. Per l'editor, una parola è definita come una stringa di caratteri compresa tra due delimitatori. Un delimitatore di parola è uno dei seguenti caratteri: blank,

return, inizio e fine (prima del CTRL-C) dell'inserimento in corso. Il trattino non è riconosciuto come separatore di parole.
I paragrafi sono separati dai seguenti delimitatori di paragrafi: una linea bianca (seguita dal Return), inizio o fine del file e una linea che comincia con il comando C(ommand character (cfr. *Ambiente*).

*AGGIUSTAMENTO AUTOMATICO E USO DEI MARGINI

Il margine sinistro è gestito con l'aggiustamento automatico sotto il primo carattere diverso dal blank della linea sopra. Il margine destro è gestito tramite l'inserimento effettuato. Non c'è controllo per i caratteri situati alla destra del cursore tranne un punto esclamativo nell'80-esima colonna se la linea supera gli 80 caratteri.
Il cambio di margine si effettua come segue: spazio per aumentare il margine, ← per diminuirlo, CTRL-Q per annullarlo. Questi caratteri servono solo se vi trovate sul primo carattere di una linea.

*NESSUN FORMATTAMENTO DEL TESTO

Sta a voi gestire i margini a sinistra e a destra, i paragrafi, ecc. I caratteri possono essere aggiunti in qualsiasi punto dello schermo. Se superate la 71-esima colonna sentirete un suono e se superate l'80-esima colonna vedrete un punto esclamativo su di essa. Questi sono gli unici controlli eseguiti.

• Cancellazione di un testo in un file

Il comando D(elete dell'editor ci consente di entrare nella modalità di soppressione di un testo. Vedrete apparire sullo schermo la seguente linea:

>DELETE: < > <MOVING COMMANDS> <EXT>
TO DELETE, <ESC> TO ABORT

Innanzitutto, il cursore deve essere situato sulla posizione a partire dalla quale deve iniziare la cancellazione. Se volete cancellare dei caratteri andando verso destra, il cursore deve trovarsi sul primo carattere da eliminare. Se vi spostate verso sinistra, il cursore dev'essere sul carattere che si trova subito a destra del primo carattere da eliminare.
Man mano che si procede con la cancellazione, i caratteri vengono soppressi e cancellati dallo schermo. Comunque, facendo l'operazione inversa, vedrete riapparire i caratteri eliminati fino alla posizione iniziale del cursore la quale è stata conservata in memoria.
Quando avete finito la vostra manipolazione, dovete battere CTRL-C per memorizzarla definitivamente. Se, invece, desiderate annullare le modifiche, dovete battere ESC. In entrambi i casi uscirete dal modo di soppressione di testi e il cursore verrà posizionato all'inizio.
Tutti i caratteri compresi tra il cursore e la sua posizione iniziale vengono memorizzati per poter essere copiati quando uscite dalla soppressione di testo annullando o meno il comando (<ESC> o <ETX>). In questo modo, se avete fatto un errore, potete recuperare i caratteri e rimetterli a posto. Potete anche copiare i caratteri in vari punti del testo oppure

trasferirne una zona. Se provate a cancellare un testo maggiore di quello che può contenere il buffer, quando battere CTRL-C per registrare la cancellazione vi compare il seguente messaggio:

THERE IS NO ROOM TO COPY THE DELETION.
DO YOU WISH TO DELETE ANY WAY? (Y/N)

In entrambi i casi, il buffer rimane nello stato in cui si trovava prima del comando D(elete. Se rispondete Y, il testo viene distrutto. Questa domanda non viene posta se tentate di copiare il testo situato nel buffer di copia. Riceverete il messaggio:

ERROR: NO ROOM. PLEASE PRESS
<SPACEBAR> TO CONTINUE.

- **Sostituzione di caratteri**

Se volete portarvi al livello sostituzione di caratteri, premete X quando siete al livello comando dell'editor, e apparirà la seguente linea:

>X(CHANGE: TEXT <BS> A CHAR <ESC>
ESCAPES; <ETX> ACCEPTS

A questo punto potete:

- sostituire il carattere sul quale è posizionato il cursore premendo un nuovo carattere; il cursore si sposterà di uno spazio verso destra;
- premere il carattere <BS> (freccia a sinistra), che fa riapparire il carattere che precedeva la modifica;
- premere ESC, che annulla le modifiche effettuate e fa ricomparire i caratteri modificati dopo essere entrati nella sostituzione;
- premere CTRL-C, che memorizza definitivamente le modifiche effettuate.

Esempio: supponiamo vogliate sostituire la linea

WRITE ("SALUT");

con la linea

WRITE ("HELLO");.

Per eseguire questa modifica seguite il procedimento:

- posizionate il cursore sulla S di "SALUT";
- digitate XHELLO;
- battete CTRL-C per rendere definitiva la modifica oppure ESC per annullarla.

- **Cancellazione del testo compreso tra la posizione corrente del cursore e l'inizio dell'ultimo testo inserito, sostituito o trovato (ZAP)**

Questo comando viene attivato battendo Z(AP al livello comando dell'editor. È utile ad esempio in questo caso:

WRITELN('MELA');

- posizionamento del cursore alla fine di MELA;
- comando I APPLE CTRL-C

WRITELN('MELAAPPLE');

- posizionamento del cursore sulla M di MELA;
- comando Z(AP

WRITELN('APPLE');

Se tentate di cancellare più di 80 caratteri alla volta, compare il seguente messaggio d'errore:

WARNING! YOU ARE ABOUT TO ZAP MORE THEN 80 CHARS, DO YOU WISH TO ZAP? (Y/N)

Se digitate N, il comando viene ignorato; se digitate Y compare il messaggio d'errore:

THERE IS NO ROOM TO COPY THE DELETION.
DO YOU WISH TO DELETE ANYWAY? (Y/N)

Se battete Y il comando viene eseguito senza copiare i caratteri cancellati nel buffer di copia.

• Copia di un testo in un file

Quando digitate il comando C al livello comando dell'editor, entrate nella copia dell'editor e appare sullo schermo la seguente linea:

>COPY: B(UFFER F(ROM FILE <ESC>

A questo punto potete inserire prima del cursore sia il testo che si trova nel buffer di copia sia un file esterno.

Copia di buffer. Battendo B, il testo contenuto nel buffer interno viene immediatamente inserito prima del cursore. Al termine della copia, il cursore viene posto sul primo carattere del testo inserito dal buffer. A differenza dell'inserimento, l'impaginazione del testo inserito non viene modificata dai margini.
Il comando C(opy del Buffer è utilizzato per copiare un testo appena inserito o per copiare un testo che viene spostato all'interno del file (comandi D(elete e C(opy). Il buffer interno viene modificato dai comandi dell'editor I(nsert, D(elete e Z(ap.

Copia di un file esterno. Digitando F(ROM FILE al livello copia dell'editor, vedete visualizzarsi il messaggio seguente:

>COPY: FROM WHAT FILE MARKER, MARKER?

Potete far inserire prima del cursore il contenuto di un file situato sul disco del vostro file attuale. Per farlo, dovete indicare il nome di un file contenuto in un disco presente nel drive; il suffisso .TEXT viene automaticamente

aggiunto alla fine del nome del file; non occorre precisarlo. Se volete indicare un nome di file senza suffisso, dovete battere un punto (.) dopo la specificazione completa del file. Se battete Return, il file specificato viene interamente inserito nel vostro file. È possibile introdurre nel vostro testo anche solo una parte del file; per farlo dovrete aver definito dei contrassegni di riferimento nel file incluso (cfr. Comando Set). Indicate i contrassegni inquadrando la parte di file che volete includere.
Se il nome del file non è seguito dai contrassegni, viene inserito tutto il file.
Se fate seguire il nome del file dal contrassegno, il file viene inserito dall'inizio fino alla prima occorrenza del contrassegno; se il nome del file è seguito dal contrassegno, viene inserita la parte di file situata dopo di esso.
Se specificate due contrassegni, viene inserita nel vostro file, solo la parte di file compresa tra i due contrassegni.
Una volta effettuato l'inserimento del file, il cursore si posiziona sul primo carattere copiato e appare sullo schermo il seguente messaggio:

> BE SURE ORIGINAL SYSTEM.EDITOR DISK IS IN THE SAME DRIVE: RETURN TO CONTINUE

Questo messaggio serve soprattutto a ricordare agli studenti dell'APPLE II monodrive di riinserire il disco di caricamento.
Se il vostro file non può contenere tutto il testo da inserire, il text editor ne conserva il più possibile e visualizza il messaggio:

> ERROR: BUFFER OVERFLOW. PLEASE PRESS <SPACEBAR> TO CONTINUE.

Premendo lo spaziatore, verrà fatta la copia di quello che si trova nel buffer, ma non del testo per intero. Prendete le vostre precauzioni per correggerla e dividete in due parti il vostro file.

Esempio: Supponiamo che esista un programma P1.TEXT e che vogliate modificarlo per ricavarne il programma P2.TEXT conservando la versione originale. Potete seguire il seguente procedimento:

- editare P2.TEXT;
- battere i comandi C(opy, F(ROM FILE e APPLE4:P1;
- il file P2.TEXT contiene, a questo punto, P1 il quale può essere modificato come desiderate.

2.5.3 Comandi per modificare la posizione del cursore

• Posizionamento del cursore

Il comando J(ump consente di posizionare il cursore:

- all'inizio del file;
- alla fine del file;
- su un dato contrassegno (cfr. par. 2.5.5).

Questo comando viene attivato premendo J al livello comando dell'editor. Sullo schermo compare:

>JUMP: B(EGINNING E(ND M(ARKER <ESC>

Se battete B posizionate il cursore in cima al file e viene visualizzata la prima pagina di testo.
Se battete E posizionate il cursore alla fine del file e viene visualizzata l'ultima pagina di testo.
Battendo M viene visualizzata la linea:

JUMP TO WHAT MARKER?

Dovete specificare un contrassegno. Se l'editor trova il contrassegno nel file vi si posiziona sopra e visualizza la pagina di testo corrispondente; altrimenti, ottenete il seguente messaggio d'errore:

ERROR: NOT THERE. PLEASE PRESS <SPACEBAR>
TO CONTINUE.

e il cursore non si sposta.

• Cambio della pagina di testo visualizzata

Il comando P(age dell'editor sposta il cursore di una pagina di testo nella direzione attiva (cfr. par. 2.5.1). Il cursore viene posizionato all'inizio di una linea. Per spostarvi di più pagine alla volta, potete utilizzare un fattore di ripetizione.

2.5.4 Comandi per lavorare su stringhe di caratteri

• Introduzione

Esistono due possibilità di lavoro sulle stringhe di caratteri:

- lavoro sui caratteri che si susseguono;
- lavoro sui caratteri riuniti in parole.

Per esempio, secondo la prima possibilità, la stringa "sa" verrà trovata nella parola casa; con la seconda possibilità la stessa stringa non viene trovata. La scelta del modo di lavoro dipende dall'ambiente.

• Ricerca di una stringa di caratteri

Il comando F(ind consente di cercare in un file le occorrenze di una stringa di caratteri. Per accedervi, vi basterà premere F al livello comando dell'editor. Vedrete comparire il messaggio:

>FIND 1: L(IT <TARGET> - > (se lavorate per parole)

o

>FIND 1: T(OK <TARGET> - > (se lavorate per caratteri)

Potete passare da un modo di lavoro all'altro digitando T(OKEN per lavorare per parole o L(ITTERAL per lavorare per caratteri.

Quando compare uno dei due messaggi, potete digitare:

- L o T per cambiare modo di lavoro;
- la stringa di caratteri da cercare, compresa tra due caratteri delimitatori uguali.

Esempio:

'PAROLA'
/WRITE/

Non appena avete battuto il secondo carattere delimitatore, il cursore si posiziona sulla fine della prima occorrenza della stringa richiesta.
Potete cercare in un file la nona occorrenza di una stringa di caratteri, specificando, quando battete il comando F, un fattore di ripetizione (numero intero compreso tra 0 e 3999).

Esempio: supponiamo che abbiate il seguente file:

```
BEGIN
WRITE X;
PAGE;
WRITE ('IL RISULTATO OTTENUTO È' RES(X))
END.
```

Posizioniamoci all'inizio del file battendo il comando JB (Jump Beginning).
Precisiamo in seguito, la direzione di spostamento + e digitiamo:

JB + 2F/WRITE

o

JB + F/WRITE/

nel primo caso vedete apparire il messaggio:

<FIND 2: L(IT <TARGET> - > /WRITE/

e il cursore si posiziona sul primo carattere che segue la E della seconda occorrenza di WRITE.
Nel secondo caso vedete apparire il messaggio:

<FIND 1: L(IT <TARGET> - > /WRITE/

e il cursore si posiziona sullo spazio che c'è tra X e E alla fine della prima occorrenza di WRITE.

1. *Se l'editor non trova alcuna occorrenza nella direzione di spostamento del modo di lavoro desiderato, visualizza il seguente messaggio d'errore:*

ERROR: PATTERN NOT IN THE FILE. PLEASE PRESS <SPACEBAR> TO CONTINUE

e non cerca la stringa nell'altra direzione.
2. *Tutti i caratteri, tranne lettere e cifre, possono essere usati come caratteri delimitatori. Se vi dimenticate di precisarli ottenete il seguente messaggio:*

ERROR: INVALID DELIMITER. PLEASE PRESS
<SPACEBAR> TO CONTINUE.

Riprovate con un delimitatore corretto.
3. *Se volete bloccare il comando* F(IND *dovete battere* <ESC>.
4. *Battendo* S *come stringa da cercare* (Target) *indicate all'editor di cercare la prossima occorrenza dell'ultima stringa cercata.*
Ad esempio, è equivalente digitare:

JB + 2F/WRITE/

e

JB + F/WRITE/FS

In entrambi i casi, il cursore si posiziona sul carattere che segue il secondo WRITE *del file.*

• Sostituzione di una stringa di caratteri

Il comando R(eplace consente di sostituire le occorrenze di una stringa di caratteri con un'altra stringa di caratteri. Per entrare nel modo di sostituzione di stringhe di caratteri, vi basta premere la lettera R al livello comando dell'editor. Potete usare due modi di funzionamento:

- per carattere (o litteral);
- per parola (o T(oken).

La scelta del modo di funzionamento dipende dall'ambiente (cfr. par. 2.5.5). Esso può essere modificato con il comando R(eplace. A seconda del modo di lavoro otterrete sullo schermo il seguente messaggio:

>REPLACE n: L(IT V(FY <TARG> <SUB> - >
(per parola)

o

>REPLACE n: T(OK V(FY <TARG> <SUB> - >
(per carattere)

n è il fattore di ripetizione del comando.
Potete scegliere tra:

- cambiare il modo di lavoro per il comando;
- indicare se volete specificare la vostra scelta ad ogni occorrenza di sostituzione (opzione V(FY);
- specificare la stringa d'origine e quella di sostituzione. Ognuna di queste stringhe deve essere indicata tra due delimitatori, (esempio /WRITE/ / WRITELN/).

Non appena avete battuto il secondo delimitatore della stringa di sostituzione, l'editor cerca nella direzione di spostamento (cfr. par. 2.5.1) le prime n occorrenze della stringa da sostituire e sostituisce loro la nuova stringa, dopo la conferma nel caso abbiate utilizzato l'opzione Verify. Il cursore si posiziona alla fine dell'n-esima occorrenza della stringa da sostituire.

Se l'editor non riesce a trovare, nella direzione di spostamento, le n occorrenze richieste, non continua dall'altra estremità del file e visualizza il messaggio d'errore:

> ERROR: PATTERN NOT IN THE FILE. PLEASE PRESS <SPACEBAR> TO CONTINUE.

L'editor può, ad ogni occorrenza della stringa, chiedere conferma del comando di sostituzione. Per utilizzare questa possibilità, è sufficiente premere V prima della stringa da sostituire. L'opzione è attivata anche se non compare nulla. Ad ogni occorrenza della stringa vedrete apparire il messaggio:

> >REPLACE: <ESC> ABORTS, 'R' REPLACE, ' ' DOESN'T

A questo punto, se battete R avviene la sostituzione della vecchia stringa con la nuova, se battete uno spazio l'editor passa o alla prossima occorrenza della stringa o al livello comando, se premete ESC il comando viene abbandonato.
Potete conservare, tra due chiamate del comando R(eplace, le stringhe indicate:

S/nuova stringa/	stesso comando con la stessa vecchia stringa
/vecchia stringa/S	stesso comando con la stessa nuova stringa
SS	stesso comando

2.5.5 Comandi per il posizionamento dell'ambiente (margini, contrassegni, formattamento...)

• **Spostamento dei caratteri su una linea A(djust**

Premere A al livello comando dell'editor provoca la visualizzazione del messaggio:

> >ADJUST: L(JUST R(JUST C(ENTER <LEFT, RIGHT, UP, DOWN-ARROWS> <ETX> TO LEAVE

Questo comando consente di spostare una linea già digitata di una posizione verso sinistra (←), verso destra (→) o di centrarla. La linea può essere spostata, verso sinistra, fino alla prima colonna; nell'altro senso, i caratteri non visualizzabili (che superano le 80 colonne) sono caratterizzati da un punto esclamativo.
Per registrare definitivamente la posizione della linea, dovete battere CTRL-C; altrimenti, battendo Esc annullerete la modifica.
È possibile spostare, in una volta, tutto un insieme di linee. Per farlo è sufficiente mettere a posto una delle linee e posizionarsi sulle altre usando CTRL-O (↑) o CTRL-L (↓). Queste ultime saranno allineate con la prima.
Ad esempio, se avete il seguente programma:

```
BEGIN;
    IF A>B
    THEN WRITE(A)
    ELSE WRITE(B);
    C=B;
END;
```

e battete i comandi A←CTRL-L CTRL-L CTRL-C, ottenete il seguente nuovo programma:

```
BEGIN
   IF A>B
   THEN WRITE(A)
   ELSE WRITE(B);
   C:=B;
END;
```

Le prime tre linee sono state spostate di una posizione verso sinistra.

• **Posizionamento dell'ambiente**

Utilizzando successivamente i comandi S(et e E(NVIRONMENT, vedete comparire sullo schermo il seguente messaggio:

```
>ENVIRONMENT: OPTIONS <ETX> OR <SP> TO LEAVE
  A(UTO INDENT              aggiustamento automatico (cfr. par. 2.5.2)
  F(ILLING          FALSE   uso dei margini (cfr. par. 2.5.2)
  L(EFT MARGIN      0       margine sinistro
  R(IGHT MARGIN     79      margine destro
  P(ARA MARGIN      5       margine paragrafo
  C(OMMAND CH               contrassegno di inizio paragrafo
  T(OKEN DEF        TRUE    ricerca delle stringhe per parola (cfr. par.
                            2.5.4)
7440 BYTES USED, 12016 AVAILABLE
```

Per cambiare una delle caratteristiche, è sufficiente battere la lettera indicata seguita dal valore desiderato.

Esempi:

- soppressione dell'aggiustamento automatico delle linee

  (SE)AF

- uso del margine:

  (SE)FT

- margine sinistro:

  (SE)L10

• Posizionamento dei margini di un paragrafo

Quando l'opzione "uso dei margini" è vera (F. True) e l'opzione "aggiustamento automatico" è falsa (A False), il comando M(argin consente di inquadrare nei nuovi margini, il paragrafo sul quale si trova il cursore.
Un paragrafo è un insieme di linee separato dalle altre da una linea bianca, dal carattere di comando o dalla fine del file.

• Posizionamento dei contrassegni

Avete la possibilità di introdurre in un file editabile (.TEXT) fino a dieci contrassegni, i quali potranno essere utilizzati dai comandi J(UMP e C(OPY.
Per posizionare un contrassegno, digitate i comandi S e M nell'ordine dato. Apparirà il seguente messaggio:

SET WHAT MARKER?

Questa domanda vi chiede di specificare il contrassegno da mettere nella posizione attuale del cursore. Il contrassegno può essere composto al massimo da otto caratteri. Se il nome indicato corrisponde ad un vecchio contrassegno, la posizione precedente viene persa. Se tentate di aggiungere un undicesimo contrassegno, compare il seguente messaggio:

MARKER OVFLW. WHICH ONE TO REPLACE?
0) nome 0
1) nome 1
.
.
.
9) nome 9

Battendo un numero, viene sostituito il contrassegno corrispondente. Questa è l'unica maniera di eliminare un contrassegno.

2.5.6 Uscita dell'editor

Il comando Q(UIT permette di uscire dall'editor. Sullo schermo compare il messaggio:

```
>Q(UIT:
 U(PDATE THE WORKFILE AND LEAVE
                    (memorizzazione del testo nel workfile)
 E(XIT WITHOUT UPDATING
                    (uscita dall'editor senza memorizzazione)
 R(ETURN TO THE EDITOR WITHOUT UPDATING
                    (ritorno all'editor)
 W(RITE TO A FILE NAME AND RETURN
                    (memorizzazione del testo in un nuovo file)
 S(AVE WITH THE SAME NAME AND RETURN
                    (memorizzazione del testo nello stesso file)
```

Per far eseguire ciò che avete scelto premete una delle lettere V, E, R, W, S.

2.6 L'ASSEMBLER 6502

L'assembler 6502 fornito col linguaggio Pascal è un macro-assembler che consente di scrivere delle procedure e delle funzioni interfacciabili con un programma Pascal o Fortran. Consente, inoltre, di effettuare l'assemblaggio condizionale.
Le pseudoistruzioni utilizzabili sono le seguenti:

- MACRO
  .
  :
- ENDM

definizione di macroistruzioni

- IF
- ELSE
- ENDC

assemblaggio condizionale

- PROC
- FUNC
- END

definizione di una procedura o di una funzione

- ASCII — stringa di caratteri
- BYTE — byte
- WORD — parola
- BLOCK — blocco dati
- ORG — indirizzo di partenza
- EQU — etichetta
- ABSOLUTE — assemblaggio assoluto
- INTERP — assemblaggio relativo

riservazione di zone di memoria e definizione di etichette

- CONST — definizione di costanti
- PUBLIC — variabile globale (Pascal e assembler)
- PRIVATE — variabile locale (assembler)

- DEF — variabile globale (due s.p. assembler)
- REF — variabile esterna (due s.p. assembler)

- LIST
- NOLIST
- MACROLIST
- NOMACROLIST
- PAGE
- TITLE
- PATCHLIST
- NOPATCHLIST

direttive di presentazione del listing

Questo assembler è potente. Se siete dei principianti, è meglio che cominciate con l'assembler LISA2.5TM o col miniassembler.
L'uso dell'assembler con il Pascal è utile per colmare le lacune di questo linguaggio per quel che riguarda l'accesso alla memoria.

2.7 IL LINKER

Consente di raggruppare programmi Pascal, Fortran e assembler per creare dei programmi eseguibili. Inoltre, consente la creazione e l'uso delle biblioteche di sottoprogrammi (APPLESTUFF, TURTLEGRAPHICS, ...).
Il collegatore permette di risolvere il problema dei riferimenti non definiti (PROC, FUNC, GLOBAL, PUBLIC, REF, DEF...) nelle diverse parti del programma. Esso crea un file in P-code interpretabile dal sistema UCSD.

2.8 LINGUAGGI DISPONIBILI CON LA SCHEDA DEI LINGUAGGI

2.8.1 Pascal

Il Pascal dell'APPLE II è il Pascal UCSD modificato in modo da poter utilizzare le possibilità grafiche ad alta risoluzione. Il risultato della sua compilazione è nel P-code accettato dal sistema UCSD; il Pascal consente di eseguire i programmi più rapidamente del Basic. Le possibilità di programmazione strutturata, di struttura di dati, ne fanno un linguaggio potente e piacevole da utilizzare.
La biblioteca di sottoprogrammi TURTLEGRAPHICS permette un facile accesso ai programmi di grafica a colori ad alta risoluzione dell'APPLE II.
Le funzioni e le procedure permettono all'utente di scegliere un colore, di spostare e muovere il cursore, di specificare i margini delle finestre visive e di colorarle, di copiare sullo schermo una tabella di dati situata in memoria e di scrivere dei caratteri in zona grafica. Esso aggiunge le seguenti procedure e funzioni:

INITTURTLE : passaggio nel modo grafico ad alta risoluzione di tutto lo schermo con sbiancamento dello schermo e posizionamento del colore trasparente e del cursore al centro dello schermo

GRAFMODE : passaggio nel modo grafico ad alta risoluzione senza sbiancamento né inizializzazione

TEXTMODE : ritorno al modo testo

VIEWPORT : definizione di una finestra grafica

PENCOLOR : posizionamento del colore del cursore

FILLSCREEN	: posizionamento del colore della finestra grafica
MOVETO	: spostamento del cursore verso il punto indicato
TURTLEX	: ottenimento della coordinata X del cursore
TURTLEY	: ottenimento della coordinata Y del cursore
TURNTO	: rotazione del cursore fino all'angolo specificato (assoluto)
TURN	: rotazione del cursore dell'angolo specificato (relativo)
TURTLEANG	: angolo con l'orizzontale "del cursore"
MOVETO	: spostamento del cursore della lunghezza indicata nella direzione fissata da TURN (o TURNTO)
SCREENBIT(X,Y)	: funzione booleana vera se il punto di coordinate X,Y non è di colore nero
DRAWBLOCK	: tracciato di una figura definita punto per punto
WCHAR	: visualizzazione di un carattere nella zona grafica
WSTRING	: visualizzazione di una stringa di caratteri nella zona grafica

Le possibilità di gestione degli Input/Output su monitor sono incluse nella biblioteca di sottoprogrammi chiamata APPLESTUFF. Le procedure e funzioni offerte sono le seguenti:

NOTE (frequenza, durata)	: emissione di suoni
KEYPRESS	: test per la battuta di un carattere su tastiera
PUDDLE(numero)	: lettura del valore indicato per una manopola di gioco
BUTTON(numero)	: test per saper se il pulsante della manopola è schiacciato
RANDOM RANDOMIZE	: generazione di numeri aleatori

Il Pascal dell'APPLE II consente di dividere i programmi in più files. Ad esempio, potete dividere tra più persone, in diverse mansioni, lo sviluppo di un software evoluto. Più è piccola la dimensione dei programmi, più la realizzazione e i tests sono facilitati, e, inoltre, rischiate meno "disastri" (gestione di files...).
Il vantaggio di questo linguaggio è dimostrato dal fatto che i linguaggi Fortran, Pilot, Logo, presentati nella prossima sezione, sono stati scritti in Pascal.

2.8.2 Fortran

Il Fortran è un linguaggio concepito in particolar modo per lavori matematici, di ingegneria e per lavori scientifici. Esistono numerose biblioteche di sottoprogrammi FORTRAN adattabili all'APPLE II. Il Fortran dell'APPLE è il sottoinsieme ANSIX3-9-78 del Fortran 77 con qualche miglioramento, come l'accesso alle possibilità grafiche ad alta risoluzione, alle possibilità sonore e alle manopole di gioco.
Inoltre, un'applicazione sull'APPLE II può essere realizzata parte in Fortran e parte in Pascal; in questo modo, si ottimizza la velocità dei calcoli numerici pur programmando il resto dell'applicazione in modo strutturato.
Il compilatore Fortran traduce i programmi in P-code. Li potrete far eseguire su qualunque macchina munita del sistema UCSD.
Grazie alla sua adattabilità e al suo carattere evoluto, il Fortran sarà per voi un importante fattore nella riduzione dei "costi" di programmazione.

2.8.3 Logo

Il Logo è un linguaggio evoluto, sviluppato dal professor Seymour Papert del M.I.T., per insegnare ai bambini, divertendoli, in un primo tempo la geometria e in seguito la programmazione. Offre numerose possibilità grafiche ad alta risoluzione simili a quelle della biblioteca TURTLEGRAPHICS già presentata. Il cursore si presenta sotto forma di una freccia (→) orientabile in tutte le direzioni e che può spostarsi da un punto all'altro tracciando o meno la retta seguita. Offre diversi colori e vi dà la possibilità di disegnare figure sullo schermo. Le nozioni di cicli di programmi e di sottoprogrammi appaiono in seguito, per evitare di scegliere sempre le stesse istruzioni.

2.8.4 Pilot

L'Apple Pilot è un sistema di sviluppo di programmi d'insegnamento assistito da calcolatore (EAD). Oltre alle possibilità del linguaggio "COMMON PILOT", questo linguaggio offre la grafica a colori, gli effetti sonori e un editor di giochi di caratteri per presentare le lezioni con parole, disegni e suoni.
Il linguaggio Apple Pilot ha due modi: Autore e Lezione. Nel primo modo, il professore crea le lezioni e le registra su disco; nel secondo modo, lo studente segue la lezione del professore e il sistema può registrare il suo livello, i suoi tempi di risposta, ecc.
Citiamo, come esempio, i seguenti comandi "Autore":

creare/editare delle lezioni
creare/editare dei grafici

creare/editare dei rumori
creare/editare dei caratteri definiti
selezionare un disco
copiare un disco

Il linguaggio Apple Pilot è stato scritto in Pascal; non necessita della scheda dei linguaggi.

CAPITOLO 3

La Softcard Z80

3.1 PRESENTAZIONE DELLA SOFTCARD

La softcard Z80 è una scheda d'espansione creata dalla società americana Microsoft. Questa scheda, contente un microprocessore Z80 di Zilog, consente di estendere le possibilità dell'APPLE II ai software scritti per lo Z80 e, in particolar modo al sistema operativo CP/M e all'interprete MBASIC della Microsoft.
Questo capitolo spiega, dopo una descrizione materiale della Softcard, la struttura del CP/M e le possibilità del BASIC e degli altri linguaggi offerti.
L'uso della Softcard necessita di 48 K bytes di memoria RAM e di almeno un drive. Essa è inseribile negli slots di espansione che vanno dal numero 1 al 7, è compatibile con la scheda di espansione accettata della scheda dei linguaggi.

3.2 STRUTTURA MATERIALE DELLA SOFTCARD

Una volta installata la Softcard, l'APPLE II può funzionare in due modi diversi:

- il modo 6502 di funzionamento normale dell'APPLE dove la Sofcard è disinnestata dall'APPLE (microprocessore Z80 fermato dal segnale HALT e Softcard separata dal bus dell'APPLE mediante dei buffers a tre stati in alta indipendenza);
- il modo Z80 di funzionamento della Softcard dove il microprocessore 6502 assicura solo il refresh delle memorie dinamiche (accesso alla memoria del 6502 e dello Z80 alternati).

Il passaggio da un modo di funzionamento all'altro si effettua indirizzando la Softcard (indirizzi corrispondenti allo slot nel quale è installata la Softcard).

Questo passaggio serve in particolar modo a far garantire dal 6502 le operazioni di Input/Output mentre lo Z80 garantisce le operazioni interne (cfr. par. 3.3.4 *Chiamate di sottoprogrammi* 6502).
La Softacard comprende:

- Dei buffer a tre stati che permettono la comunicazione tra i bus e Z80.
- La logica necessaria per ottenere un orologio per lo Z80 dalle fasi dell'orologio 6502. La frequenza ottenuta è di 2 MHz.
- Un modificatore d'indirizzo che permette di adattare la memoria dell'APPLE alla struttura classica CP/M.

La scheda di memoria ottenuta è la seguente:

Indirizzi 6502	**Indirizzi Z80**	**Contenuto**
$0800-$0FFFF	0F800-0FFFF	Unità di controllo di I/O
$0400-$07FF	0F400-0F7FF	Memoria di schermo
$0200-$03FF	0F200-0F3FF	Buffer I/O e indirizzi speciali
$0000-$01FF	0F000-0F1FF	Pagina 0 e pagina 1 (stack 6502)
$C000-$CFFF	0E000-0EFFF	Mapping degli Input/Output ROM Schede di espansione
$FFFA-$FFF	0DFFA-0DFF	Vettori 6502 RESET, BREAK, NMI
$F800-$FFFF	0D800-0DFF9	ROM Autostart
$D000-$F7FF	0B000-0D7FF	ROM APPLESOFT, Integer Basic RAM scheda dei linguaggi
$1000-$BFFF	0000-0AFFF	RAM continua

- La Softcard comprende anche la logica necessaria per garantire il refresh delle memorie RAM dinamiche e del microprocessore 6502. Essa mette il 6502 in stato di attesa (segnale RDY basso) e gli cede il controllo del bus mentre lo Z80 sta decodificando l'istruzione che dovrà eseguire.
- Quattro "switchs" che hanno i seguenti ruoli:

  - S1 posizionato — nessuna modifica d'indirizzo
    non posizionato — modifica d'indirizzo.
  - S2 posizionato — autorizzato l'accesso diretto alla memoria Z80.
  - S3 posizionato — considerata l'interruzione non mascherabile.
  - S4 posizionato — considerata l'interruzione mascherabile.

Il lettore interessato al funzionamento della Softcard dovrà rifarsi ai manuali tecnici di microprocessori Rockwell 6502 e Zilog Z80.

3.3 IL SISTEMA OPERATIVO CP/M

3.3.1 Introduzione al CP/M

Il CP/M è un sistema operativo scritto per i microprocessori Intel 8080 e Zilog Z80. Essendo molto diffuso, questo sistema dà accesso ad un vasto software:

- software di base:
  - text editor;
  - linguaggi evoluti: Pascal, Basic, Cobol, Fortran, PL/1, LISP;
  - strumenti di debugger e disassemblaggio;
- software di applicazione orientato verso il trattamento di testi, ...

Grazie alla sua struttura interna, i programmi scritti su una macchina sono adattabili su un'altra macchina che fa uso del CP/M.

3.3.2 Ambiente del CP/M sull'APPLE

Il CP/M è compatibile con tutte le schede accettate dalla scheda dei linguaggi.
Tuttavia, per le altre schede di espansione, lo spazio è imposto.
Slot 0: scheda dei linguaggi, APPLESOFT, Integer Basic.
Slot 1: scheda per l'interfaccia stampante (parallela o seriale) (periferica logica LST:).
Slot 2: scheda per l'interfaccia degli Input/Output generalizzati (periferiche logiche RDR: e PUN:):

- scheda per l'interfaccia cassetta;
- scheda per l'interfaccia lettore/perforatore nastri;

o qualunque altra scheda di Input/Output.

Slot 3: scheda per l'interfaccia Input/tastiera (periferica logica CON:).
*Nel caso in cui una scheda sia presente in questo slot, lo schermo standard dell'*APPLE *non viene più utilizzato.*
Slots 4, 5, 6: unità di controllo dei drives;
4: drives E e F,
5: drives C e D,
6: drives A e B.
Slot 7: scheda di espansione di qualunque tipo.

Esempio: scheda per l'interfaccia con un televisore a colori.

La scheda Z80 può essere inserita in qualunque slot libero.
Grazie alla struttura interna del CP/M, l'utente della Softcard può adattarlo a una qualsiasi scheda di espansione.

3.3.3 Uso del CP/M

- **Generalità**

Il CP/M (Control Program/Microprocessors) è composto da numerosi piccoli programmi la cui funzione è quella di gestire gli Input/Output con l'utente e di garantire il funzionamento dei dischi. Il CP/M offre all'utente l'accesso a numerosi sottoprogrammi di Input/Output e facilita, in questo modo, la scrittura di software in linguaggio assembler 8080.
La maggior parte dei programmi del CP/M sono programmi su disco, accessibili e modificabili dall'utente.
Nelle prossime sezioni, presenteremo brevemente i comandi del CP/M. Il lettore interessato, per saperne di più, potrà leggere il manuale di riferimento del CP/M.

- **Comandi residenti in memoria**

Sono i seguenti sei:

DIR : visualizzazione del contenuto di un disco.
ERA : distruzione di un file.
REN : modifica del nome di un file.
SAVE: memorizzazione del contenuto di una zona di memoria in un file.
TYPE: visualizzazione del contenuto di un file sullo schermo.
USER: cambio del contesto secondo il numero dell'utente.

- **Comandi residenti su file**

Sui dischi forniti con la Softacard si trovano i seguenti comadi:

STAT : - modifica degli attributi di un file,
- visualizzazione del contenuto di un disco con gli attributi dei files e lo spazio occupato,
- modifica degli assegnamenti tra periferiche logiche e fisiche (cfr. par. 3.3.2).

ASM : assemblaggio di un programma.
LOAD : creazione di un comando eseguibile.
DDT : messa a punto di un programma.
PIP : manipolazioni di files (copia, concatenazione, stampa del contenuto).
ED : editing di testi.
SUBMIT: esecuzione di un comando catalogato.
DUMP : stampa del contenuto binario di un file.

3.3.4 Struttura interna del CP/M e adattamento all'APPLE II

Il CP/M è composto da tre moduli:

- il BIOS (Basic I/O System), che fornisce i sottoprogrammi necessari per gestire fisicamente i dischi e interfacciare le periferiche standards (schermo, tastiera, stampante, modem, ...);

- il BDOS (Basic Disk Operating System), che gestisce logicamente i dischi e i files;
- il CCP (Console Command Processor), che è l'interfaccia tra utente e CP/M. Usa allo scopo il BIOS e il BDOS;
- il TPA (Transcient Program Area), che è la zona riservata ai programmi degli utenti.

Vi indicheremo quì sotto gli indirizzi di questi moduli nei due casi seguenti:

- APPLE II 48 K;
- APPLE II 48 K con scheda dei linguaggi -- APPLE IIe

APPLE II 48 K		APPLE II 64 K (scheda dei linguaggi)	
(o CP/M 44 K)		(o CP/M 56 K)	
0C000H – 0A000H	BIOS	0E000H – 0D000H	BIOS
	BDOS		BDOS
9C00H – 9400H	CCP	0CC00H – 0C400H	CCP
100H	TPA (utente)	100H	TPA (utente)

Studieremo ora la struttura dei tre moduli BIOS, BDOS e CCP, il loro adattamento all'APPLE e le funzioni CP/M offerte all'utente.
Siccome gli esempi dati sono in assembler 8080, è necessario conoscere bene i comandi del CP/M e la programmazione assembler.

• Il BIOS

Presentazione. Il BIOS gestisce gli Input/Output con le schede di espansione classiche dell'APPLE II, con lo schermo e la tastiera.
Per poter offrire il CP/M all'utente, il Bios è stato adattato all'APPLE. Esso può essere adattato dall'utente ad ogni hardware specifico e a tutte le schede di espansione APPLE.
Il BIOS è composto da sottoprogrammi assembler Z80 e 6502. Riconosce le schede di espansione esistenti al momento del caricamento del sistema.
Il BIOS accetta le stesse schede di espansione della scheda dei linguaggi. Queste schede devono essere in un preciso slot:

Slot 1	stampante
Slot 2	Input/Output seriale
Slot 3	schermo-tastiera, teletype, scheda 80 colonne
Slots 4, 5, 6	unità di controllo dei dischi

Adattamento del **Bios** *a un dato hardware.* Il BIOS è adattabile a qualunque hardware modificando il blocco di configurazione degli Input/Output (I/O configuration block).
Quest'ultimo contiene le seguenti informazioni:

- configurazione delle funzioni grafiche dello schermo;
- riconfigurazione dei caratteri della tastiera;
- tavola degli indirizzi dei sottoprogrammi di Input/Output e sottoprogrammi considerati;
- tavola degli indicatori di presenza delle schede di Input/Output.

Ogni disco di sistema contiene il suo blocco di configurazione degli Input/Output che viene caricato e inizializzato (indicatori di presenza quando il sistema è lanciato).
Studieremo ora le diverse funzioni del blocco di configurazione degli Input/Output.

* FUNZIONI GRAFICHE E INDIRIZZAMENTO DEL CURSORE

Presentazione. La maggior parte dei terminali video (tra il quali lo schermo standard dell'APPLE II) può sopportare delle funzioni speciali quali l'indirizzamento diretto del cursore, lo sbiancamento dello schermo, della linea corrente o il passaggio in modo inverso dello schermo (lettere nere su fondo bianco). Queste funzioni vengono realizzate mandando sullo schermo una certa sequenza di caratteri.
I problemi di compatibilità tra i vari tipi di terminali hanno portato alla creazione della seguente tavola per la traduzione in codice:

Software (schermo emulato)/Hardware (schermo dell'APPLE)
(terminale Soroc/terminale Datamedia generalmente).

Le seguenti nove funzioni sono accettate dal BIOS:

1. sbiancamento dello schermo;
2. sbiancamento fino alla fine dello schermo;
3. sbiancamento fino alla fine della linea;
4. passaggio al modo normale;
5. passaggio al modo inverso;
6. posizionamento del cursore in alto e a sinistra;
7. indirizzamento del cursore;
8. salita del cursore in una linea;
9. spostamento del cursore di una posizione verso destra;
10. discesa del cursore di una linea;
11. spostamento del cursore di una posizione verso sinistra.

Le ultime due funzioni sono garantite in maniera standard, digitando i seguenti caratteri:

- Line feed (LF codice ASCII 10) per la funzione 10;
- Backspace (BS codice ASCII 8) per la funzione 11.

Per realizzare queste funzioni grafiche inviare sullo schermo due tipi di sequenze:

- un carattere unico;
- un carattere ASCII, preceduto da un carattere di introduzione.

Il formato della tavola per la traduzione in codice Sofware/Hardware è il seguente:

N° della funzione	Indirizzo del codice nella tavola relativa		Descrizione
	Software	**Hardware**	
	0F396H	0F3A1H	Valore da aggiungere alle coordinate X, Y per posizionare il cursore (intervallo 0-127). Se il primo bit di sinistra è 0, le coordinate devono essere inviate nell'ordine Y, X. Se il primo bit è 1 l'ordine è X, Y
	0F397 H	0F3A2H	Carattere di introduzione (0 se non occorre ad alcuna funzione)
1	0F398H	0F3A3H	Sbiancamento dello schermo
2	0F399H	0F3A4H	Sbiancamento della fine dello schermo
3	0F39AH	0F3A5H	Sbiancamento della fine della linea
4	0F39BH	0F3A6H	Passaggio dello schermo in modo normale
5	0F39CH	0F3A7H	Passaggio dello schermo in modo inverso
6	0F39DH	0F3A8H	Posizionamento del cursore in alto e a destra dello schermo
7	0F39EH	0F3A9H	Indirizzamento del cursore
8	0F39FH	0F3AAH	Salita di una linea
9	0F3A0H	0F3ABH	Spostamento del cursore di una posizione verso destra

Se una delle caselle della tavola è a zero significa che la funzione corrispondente non è più disponibile. Il carattere di introduzione è da inviare sullo schermo solo se il bit più significativo della casella è a 1.
Lo schermo standard dell'APPLE regge queste nove funzioni.

Esempio d'uso. Il fatto di usare la tavola delle funzioni grafiche consente di ottenere dei programmi indipendenti dall'harware sul quale si lavora.
Sono adottabili due posizioni:

- Si decide di utilizzare un dato schermo e si usa la parte software della tavola solo per interfacciare i programmi dell'APPLE.
La maggior parte del software scritto con il CP/M su altre macchine usa il terminale Soroc. Riempiendo la parte software della tavola con le caratteristiche del terminale Soroc e la parte hardware con le caratteristiche del terminale utilizzato e dello schermo APPLE, l'adattamento del software CP/M sull'APPLE è altamente facilitato.
- Si decide di non conoscere lo schermo considerato e si usano le funzioni grafiche leggendo il contenuto della tavola per sapere quali caratteri inviare allo schermo.

Modifica della tavola delle funzioni grafiche. La modifica della tavola per la traduzione in codice è effettuata dal programma CONFIGIO.BAS, rispondendo con 1 al menù proposto. Compare allora il contenuto della tavola seguito da un nuovo menù.

+ TERMINAL SCREEN FUNCTION DEFINITION +

FUNCTION	SOFTWARE	HARDWARE
CLEAR SCREEN	ESC *	FF
CLR TO EOS	ESC Y	VT
CLR TO EOL	ESC T	GS
LO-LITE TEXT	ESC)	SO
HI-LITE TEXT	ESC (	SI
HOME CURSOR	RS	EM
ADDRESS CURSOR	ESC =	RS
XY COORD OFFST	32	32
XY XMIT ORDER	YX	YX
CURSOR UP	VT	US
CURSOR FORWARD	FF	FS

1. SOROC IQ 120/IQ 140
2. HAZELTINE 1500/1510
3. DATAMEDIA
4. OTHER
Q. QUIT

SELECT –

Sono proposti tre tipi di terminali (Soroc, Hazeltine, Datamedia). Potete sceglierli digitando il numero corrispondente. È possibilie scegliere una quarta opzione (4) nella quale potete definire separatamente ogni funzione.

+ + SCREEN FUNCTION DEFINITION + +

1 - LEAD-IN CHARACTER
2 - CLEAR SCREEN
3 - CLR TO EOS
4 - CLR TO EOL
5 - LO-LITE TEXT
6 - HI-LITE TEXT
7 - HOME CURSOR
8 - ADDRESS CURSOR -19
9 - CURSOR UP
10 - CURSOR FORWARD
Q - QUIT

SELECT –

Una volta scelta la funzione, il programma vi chiede il carattere da inserire nella tavola e diverse risposte relative al numero della funzione.
Quando avete terminato di definire la vostra tavola, occorre memorizzarla su disco, premendo Q per ritornare al menù principale e 4 per rendere valide le vostre modifiche.

* RIDEFINIZIONE DEI CARATTERI DELLA TASTIERA

Presentazione. Alcuni software CP/M necessitano di caratteri speciali, generalmente accessibili su tastiera.
La tastiera dell'APPLE II, in questo dominio è incompleta. Questo problema è risolto con la tavola di riconfigurazione dei caratteri.
La riconfigurazione dei caratteri è effettuata dalla funzione CP/M numero 1 (cfr. par. 3.3.4).
La tavola è situata all'indirizzo 0F3ACH e consente di riconfigurare 6 caratteri. È strutturata come segue:

codice ASCII del carattere	codice ASCII del carattere riconfigurato

Tutti i caratteri della tavola hanno il loro bit più significativo a 0 tranne l'ultimo tra loro.

Modifica della tavola. La modifica della tavola si effettua utilizzando il programma CONFIGIO.BAS e rispondendo 2 al menù proposto. Sullo schermo compaiono le seguenti linee:

```
+ + KEYBOARD CHARACTER DEFINITION + +

Ctrl-K  →
Ctrl-A  →  RUB
Ctrl-B  →
Ctrl-U  →  Ctrl-I

ADD/DELETE/QUIT(A/D/Q)
```

Per aggiungere un carattere, premere A.
Per cancellare un carattere, premere D.
Per ritornare al menù principale, premere Q.
Se si è battuto A, il programma visualizza:

```
CHAR
```

Occorre digitare il carattere da modificare, il quale può essere nei seguenti formati:

- carattere;
- nome di un carattere ASCII (esempio NUL, BEL, LF, CR);
- CTRL-carattere;
- LC-carattere (LC significa lower case, cioè minuscolo);
- codice ASCII esadecimale (preceduto da BH).

Il lettore interessato, per conoscere i nomi dei caratteri e i codici, dovrà rifarsi all'allegato.
Una volta che il carattere da modificare è stato riconosciuto dal programma, questo visualizza una freccia → e l'utente deve specificare il nuovo carattere seguendo i formati suddetti.
É il medesimo criterio adottato per la cancellazione della ridefinizione di un carattere.
Una volta terminate le modifiche, occorre copiarle su disco. Rispondendo 4 *al menù principale, il blocco di configurazione degli Input/Output viene copiato sul disco di sistema* CP/M.

* ADATTAMENTO DELLE SCHEDE DI ESPANSIONE NON STANDARD

Presentazione. L'utente può adattare il CP/M a schede di espansione non standard e a software riguardante gli Input/Output. Tutti i sottoprogrammi di Input/Output sono messi in forma vettoriale in una tavola contenuta nel blocco di configurazione degli Input/Output. Normalmente, questa tavola fornisce gli indirizzi dei sottoprogrammi di Input/Output standard del BIOS, ma l'utente può modificarla facendo in modo che questa punti ai suoi programmi.
Per la realizzazione di una funzione grafica, il registro B *contiene il numero della funzione al momento della chiamata della subroutine di visualizzazione su console* (4, 5).
La struttura della tavola è la seguente:

Numero vettori	Indirizzo del vettore	Nome		Descrizione	Valore standard
1	0F380H	Stato della console		Rinvio 0FFH nel registro A se un carattere è stato digitato, altrimenti 00H	0F34AH
2	0F382H	Lettura della	1	Lettura di un carattere dalla	0F358H
3	0F384H	console	2	console e allocazione nel registro A con il bit più significativo	0AB12H
4	0F386H	Visualizzazione	1	Visualizzazione del carattere	0F35EH
5	0F388H	sulla console	2	il cui codice ASCII è contenuto nel registro C	0AC3EH
6	0F38AH	Lettura di una	1	Lettura di un carattere su	AD45H
7	0F38CH	cassetta (periferica logica RDR:)	2	cassetta (Input generalizzato), allocazione nel registro A	AD45H
8	0F38EH	Scrittura su	1	Invio del carattere contenuto	AD3FH
9	0F390H	cassetta (PUN:)	2	in C alla cassetta (Output generalizzato)	AD3FH
10	0F392H	Stampa di un	1	Invio del carattere contenuto	AD2BH
11	0F394H	carattere (LST:)	2	in C alla stampante	

Per i drivers (subroutines che gestiscono gli Input/Output) dell'utente, vengono riservate tre zone di memoria da 128 bytes.

0F200H - 0F27FH Slot 1 periferica logica LST:
0F280H - 0F2FFH Slot 2 periferiche logiche PUN: e RDR:
0F300H - 0F37FH Slot 3 periferiche logiche CRT: e TTY:

la maggior parte di Input/Output hanno un driver 6502. L'interfaccia di queste schede si effettua chiamando questi drivers (cf. infra) e scrivendo l'indirizzo della subroutine che chiama il driver 6502 nella tavola descritta prima.

* CHIAMATA DI SUBROUTINES 6502

È possibile chiamare delle subroutines assembler 6502. Infatti è facile passare dal modo di funzionamento Z80 al modo di funzionamento 6502, e viceversa.
Per chiamare un sottoprogramma 6502, l'utente deve procedere in questo modo:

- posizionare l'indirizzo della subroutine da chiamare sull'indirizzo 0F3D0H secondo l'ordine dello Zilog (cioè byte meno significativo e poi byte più significativo);
- posizionare gli argomenti da passare ai seguenti indirizzi:

Indirizzi Z80	Indirizzi 6502	
0F045H	$45	Registro A 6502
0F046H	$46	Registro X 6502
0F047H	$47	Registro Y 6502
0F048H	$48	Registro P 6502 (stati)
0F049H	$49	Registro S 6502

Questa zona di passaggio dei parametri è utilizzata dalle subroutines IOSAVE *e* IOREST *del monitor dell'*APPLE.
- *Scrittura all'indirizzo della Softcard.*
L'indirizzo della scheda è 0EN00H, *dove* N *è il numero dello slot nel quale è situata la Softcard.*
Quest'indirizzo viene riconosciuto dal CP/M *al caricamento del sistema e viene copiato all'indirizzo* 0F3DEH.
La chiamata si esegue con la seguente sequenza:

```
LHLD  0F3DEH
MOV   M, A; scrittura all'indirizzo puntato da 0F3DEH
```

La possibilità di chiamare delle subroutines 6502, serve più che altro, a interfacciare nuove schede di espansione chiamando i sottoprogrammi in ROM delle schede.

L'uso della memoria presente sulla scheda dei linguaggi è diverso a secondo del modo di funzionamento della scheda Z80.
In modo Z80 (cfr. par. 3.2), la RAM della scheda dei linguaggi è accessibile in lettura e in scrittura e corrisponde agli indirizzi $D000-$FFFF del 6502.
In modo 6502, la ROM della scheda centrale dell'APPLE è selezionata il lettura e la RAM della scheda dei linguaggi non è accessibile in lettura. Invece, ogni scrittura agli indirizzi citati viene realizzata nella RAM.

* INDICAZIONE DI PRESENZA DELLE SCHEDE DI ESPANSIONE

Al caricamento, il CP/M effettua un test per ogni slot per vedere se contiene una scheda di espansione. Il risultato ottenuto è il seguente:

0 nessuna scheda
1 scheda sconosciuta
2 unità di controllo dei dischi
3 APPLE Communication Interface
CCS 7710A Serial Interface (Interfaccia seriale)
4 APPLE High Speed Serial Interface
Videx-Videotherm
APPLE Silentype Printer Interface
5 APPLE Parallel Printer Interface (stampante interfaccia parallela)

Questi risultati sono riuniti in una tavola, all'indirizzo 0F3B9H. Il valore ottenuto per uno slot S è inserito all'indirizzo 0F3B8H + S.
Il numero delle unità di controllo dei dischi si trova all'indirizzo 0F3B8H.

• Il BDOS

Come abbiamo già detto, il BDOS è quella parte del CP/M incaricata della gestione logica dei files e dei dischi.
Un file è gestito da dei f.c.b. (file control block o blocco di controllo dei files). Ogni f.c.b. descrive 16 K bytes di un file, cioè 128 settori. Un file può utilizzare 512 f.c.b., per cui può avere una dimensione logica di 8 megabytes (la dimensione massima fisica è quella di un disco).
Un file è costituito da una sequenza di caratteri ASCII. Ogni linea termina con la sequenza carriage return 0DH e line feed 0AH. Un settore di 128 bytes può dunque contenere più linee di testo. La fine del file è contrassegnata da un CTRL-Z (1AH) o da una fine di file fisica.
Al momento dell'apertura di un file, l'utente deve fornire l'indirizzo di un f.c.b. che è stato parzialmente riempito (nome file). Il CP/M consulterà il catalogo del disco e inizializzerà il resto dei dati del f.c.b. Alla chiusura del file, l'f.c.b. in memoria verrà copiato nel catalogo.
Per default, viene riservata una zona in memoria per due f.c.b. all'indirizzo 005CH.

005CH nome del primo file
006CH nome del secondo file

È compito dell'utente copiare il nome di un secondo file in un altro f.c.b. prima dell'apertura di uno dei files.
La struttura di f.c.b. è la seguente:

dr	f1 f2 f8	t1 t2 t3	ex	s1 s2	rc	d0 d15	cr r0 r1 r2
00	01 02...08	09 10 11	12	13 14	15	16... 31	32 33 34 35

con

dr = numero del drive
0 : drive attivo per default
1 : A
2 : B
.
:
16 : P
f1...f8 nome del file (con bit più significativo a 0 0 dei fi)

t1, t2, t3 — tipo del file con bit più significativo
t1 = 1 file accessibile solo in lettura
t2 = 1 file SYS non listato da DIR

ex numero del blocco da 16 K attivo

s1, s2 — parametri interni
rc numero del settore nel blocco ex
d0...d15 parametri interni
cr prossimo settore da leggere o scrivere in un file sequenziale inizializzato dall'utente
r0, r1, r3 numero del settore al quale si accederà in accesso diretto.

Per gli Input/Output su disco, viene riservata in memoria una zona di 128 bytes. L'indirizzo di questa zona è, per default, 0080H. Questa è modificabile dall'utente utilizzando una funzione del BDOS (cfr. par. 3.3.4 Funzioni CP/M).
Il BDOS, che usa il BIOS per gli Input/Output fisici su disco, non è stato modificato per installare il CP/M sull'APPLE II.

• Il CCP

Il CCP è la parte del CP/M che dialoga con l'utente. Quando un comando viene digitato su tastiera, il CCP analizza la linea digitata:

- copia la parte della linea da analizzare col comando dell'indirizzo 080H;
- crea gli f.c.b. (cfr. BDOS) relativi ai files da manipolare col comando dell'indirizzo 05CH;

- analizza il comando digitato, controlla se si tratta di un comando interno (DIR, SAVE, TYPE, USER, REN, ERA) o di un comando esterno da caricare. Nel primo caso, chiama la subroutine corrispondente, nel secondo caso, consulta la "directory" (catalogo) del disco e carica in memoria il comando all'indirizzo 100H. Al termine del comando, rende il controllo all'utente.

Per esempio, se si digita il seguente comando:

A>MBASIC A:JEU.BAS

il CCP procederà nel seguente modo:

1. Inizializzazione
- all'indirizzo 005CH:

```
01  4A  45  55  42  41  53
    J   E   U   B   A   S
Numero drive (A:)
```

- all'indirizzo 0080H:

A: J E U.B A S

2. Caricamento e lancio del comando MBASIC.COM.

Il CCP è composto dalle seguenti parti:

- subroutines standard di accesso alle funzioni CP/M;
- subroutines per l'analisi dei caratteri in un buffer;
- subroutines per l'analisi dei comandi digitati alla tastiera;
- subroutines di caricamento e di lancio dei comandi esterni;
- subroutines corrispondenti ai sei comandi interni del CP/M;
- programma di concatenamento delle diverse parti del CCP.

Il CCP che usa i comandi del BIOS e del BDOS (funzioni CP/M) è indipendente dalla configurazione materiale. Non è stato modificato per adattare il CP/M all'APPLE II.

• Le funzioni CP/M offerte all'utente

Parecchi sottoprogrammi del CP/M sono accessibili dai programmi in linguaggio assembler dell'utente. Si classificano in due categorie:

- subroutines del BIOS;
- subroutines del BDOS.

Nel BIOS sono accessibili i seguenti sottoprogrammi:

- visualizzazione di un carattere;
- acquisizione di un carattere;
- lettura/scrittura di una cassetta;
- scrittura su stampante di un carattere;
- visualizzazione/acquisizione di un buffer;
- lettura dello stato della console (schermo/tastiera).

Sottoprogrammi accessibili nel BDOS:
- creazione, apertura, chiusura, distruzione di un file;
- lettura/scrittura dei files sequenziali e ad accesso diretto;
- modifica delle caratteristiche di un file.

L'accesso a questi sottoprogrammi si effettua in maniera standard:
- caricamento nel registro C del codice della funzione;
- chiamata della subroutine situata all'indirizzo 005H.

Eccovi la lista delle funzioni accessibili col loro numero d'ordine, i parametri di chiamata e di uscita, e una breve semantica.

Numero della funzione	Parametri di chiamata	Parametri di ritorno	Semantica
0	C=00H		Rilancio del sistema
1	C=01H	A=Codice ASCII	Acquisizione di un carattere (resta bloccata finché non viene digitato un carattere sulla tastiera) e riconoscimento dei seguenti caratteri di controllo: CTRL-I, (Tab), CTRL-S, CTRL-P, CTRL-H (Backspace), LF (line feed) CR (carriage return) (periferica CON:) cfr. par. 3.3.2
2	C=02H E=Codice ASCII del carattere da visualizzare		Visualizzazione di un carattere (CTRL-I (Tab), CTRL-P, CTRL-S ... sono trattati come sopra) (periferica CON:) cfr. par. 3.3.2
3	C=03H	A=codice ASCII del carattere letto sulla cassetta o sul nastro	Lettura di un carattere su cassetta o nastro. Resta bloccata fin quando non viene letto un carattere (periferica RDR:) cfr. par. 3.3.2
4	C=04H E=codice ASCII del carattere da scrivere		Scrittura di un carattere su cassetta o nastro (periferica PUN:) cfr. par. 3.3.2
5	C=05H E= codice ASCII del carattere da stampare		Scrittura di un carattere su stampante (periferica LST:) cfr. par. 3.3.2
6	C=06H E=0FFH (Input) o codice ASCII (Output)	A=codice ASCII o status	Input/Output diretto (i classici controlli CP/M non vengono più effettuati)

(segue)

(seguito)

Numero della funzione	Parametri di chiamata	Parametri di ritorno	Semantica
7	C=07H	A=byte di I/O	Lettura del byte di I/O (cfr. par. 3.3.4) (IOBYTE)
8	C=08H E=byte di I/O		Scrittura del byte di I/O (cfr. par. 3.3.4) (IOBYTE)
9	C=09H DE=indirizzo stringa di caratteri		Visualizzazione di una stringa di caratteri che termina con $ (i controlli CP/M della funzione 2 vengono effettuati)
10	C=0AH DE=indirizzo del buffer	Caratteri presi	Lettura di una linea dalla console Al ritorno della funzione, il buffer del CP/M ha il seguente formato: mx nc cl...cn... mx: numero massimo delle prese inc: numero dei caratteri presi sono riconosciute le seguenti funzioni: rub/del: sopprime la echo dell'ultimo carattere CTRL-C: ricaricamento del sistema se inizio linea CTRL-E: fine linea fisica CTRL-H: back-space CTRL-J: line feed } fine dell'acquisizione CTRL-M: return } CTRL-R: copia della linea corrente CTRL-X: ritorno a inizio linea
11	C=0BH	A=stato della console	A=OFFH se un carattere è stato digitato, 00H altrimenti
12	C=0CH	HL=numero di versione H=CP/M o MP/M L=versione	Lettura del numero di versione
13	C=ODH		Reinizializzazione dei dischi: - inserire tutti i dischi senza protezione scrittura, - riposizionare l'indirizzo DMA a 80H
14	C=0EH E=numero drive 0 A: 1 B: . : 15 P:		Selezione del drive attivo

(segue)

(seguito)

Numero della funzione	Parametri di chiamata	Parametri di ritorno	Semantica
15	C=0FH DE=indirizzo del f.c.b.	A=codice della directory	Apertura di un file: il BDOS cerca un file il cui nome corrisponde a quello dato nel f.c.b. Se non viene trovato alcun file A=255, se è stato trovato un file A= da 0 a 3 e il f.c.b. viene caricato in memoria
16	C=10H DE=indirizzo del f.c.b.	A=codice della directory	Chiusura di un file: Il BDOS cerca un file il cui nome corrisponde a quello nel f.c.b. Se non viene trovato alcun file A=255. Se è stato trovato un file, il f.c.b. viene copiato nel catalogo e A=0, 1, 2, 3. Le modifiche del file vengono registrate definitivamente sul disco
17	C=11H DE=indirizzo del f.c.b.	A=codice della directory	Ricerca della prima occorrenza di un file nella directory. Se non viene trovato alcun file A=255, altrimenti, il settore nel quale si trova il file nella directory viene copiato nel buffer che comincia all'indirizzo DMA e l'entrata del file viene messa all'indirizzo relativo A*32 o A=0,1,2,3
18	C=12H	A=codice della directory	Ricerca della prossima occorrenza di un file nella directory (cfr. Funzione 17)
19	C=13H DE=indirizzo del f.c.b.	A=codice della directory	Distruzione di un file=255 se non è stato distrutto alcun file, altrimenti A=0,1,2,3
20	C=14H DE=indirizzo del f.c.b.	A=codice della directory A=00H 0K < >00H fine settore	Lettura di un file sequenziale: lettura del settore numero "cr" del blocco del file. Il numero del settore dev'essere stato inizializzato dall'utente dopo l'apertura del file. Il numero di settore cr viene incrementato di un'unità e il passaggio del blocco corrente "ex" al blocco seguente è effettuata automaticamente. Il settore letto viene copiato nel buffer di indirizzo DMA
21	C=15H DE=indirizzo del f.c.b.	A=codice della directory A=00H 0K A< >00H scrittura impossibile (disco pieno)	Scrittura di un file sequenziale: Il buffer situato all'indirizzo DMA viene copiato nel settore cr. La gestione di "cr" è uguale a quella della funzione 20

(segue)

(seguito)

Numero della funzione	Parametri di chiamata	Parametri di ritorno	Semantica
22	C = 16H DE = indirizzo del f.c.b.	A = codice della directory A = 0 1 2 3 0K = 255 directory piena	Creazione di un file: Il BDOS crea il file caratterizzato dal f.c.b. È compito dell'utente verificare che due files non abbiamo lo stesso nome
23	C = 17H DE = indirizzo del f.c.b.	A = codice della directory = 0 1 2 3 0K = 255 file non trovato	Creazione di un file: Cambio del nome di un file
24	C = 18H	HL = vettore logico dei drives A = L (compatibilità con le vecchie versioni)	Lettura del vettore logico dei drives H: 15 … 8 — L: 7 … 0 il bit n. i è uguale a 1 se il drive corrispondente è attivo
25	C = 19H	A = n drive attivo 0 → A: . : 15 → P:	Lettura del numero del drive attivo
26	C = 1AH DE = indirizzo DMA		Posizionamento dell'indirizzo DMA: indirizzo di base di un buffer che serve agli I/O. È inizializzato per default a 80H
27	C = 1BH	HL = indirizzo del vettore di allocazione	In memoria, esiste un vettore di allocazione per ogni disco attivato. Questa funzione fornisce l'indirizzo del vettore di allocazione del disco attivo
28	C = 1CH		Passaggio del disco in protezione scrittura fino al prossimo ricaricamento (CTRL-C). Un tentativo di scrittura sul disco produrrà il messaggio Bdos Err on d: R/O

(segue)

(seguito)

Numero della funzione	Parametri di chiamata	Parametri di ritorno	Semantica
29	C = 1DH	HL = vettore letto	Acquisizione del vettore che indica i dischi accessibili solo in lettura (protezione in scrittura) Il bit i è = 1 se il drive corrispondente è protetto in scrittura HL 15...0 P: A:
30	C = 1EH DE = indirizzo del f.c.b.	A = codice del catalogo	Posizionamento degli attributi di un file (i.e. bits t1, t2, del f.c.b. - cfr. par. 3.3.4 Il BDOS)
31	C = 1FH	HL = indirizzo del blocco dei parametri del disco	Lettura dell'indirizzo del blocco dei parametri del disco (spazio libero,...)
32	C = 20H E = 0FFH (lettura) oppure E = numero utente	A = valore letto	Se E = 0FFH, lettura del numero utente corrente (cfr. Funzione USER), altrimenti posizionamento del numero
33	C = 21H DE = indirizzo del f.c.b.	A = codice di ritorno 00 OK 01 nessun dato scritto 03, 04, 06 } altri errori	Lettura di un file ad accesso diretto. Il CP/M legge il record indicato da r0, r1, lo copia nel buffer situato all'indirizzo DMA e posiziona il codice di ritorno
34	C = 22H DE = indirizzo del f.c.b.	A = codice di ritorno	Scrittura di un file ad accesso diretto. Il CP/M copia i dati all'indirizzo DMA nel record indicato da r0, r1 e posiziona il codice di ritorno
35	C = 23H DE = indirizzo del f.c.b.	r_0, r_1 = lunghezza	Calcolo della lunghezza del file
36	C = 24H DE = indirizzo del f.c.b.	r_0, r_1 = prossimo campo a cui accedere	Posizionamento del numero del prossimo campo da leggere/scrivere (utile, ad esempio, dopo accessi sequenziali)

Abbiamo indicato delle funzioni che vengono utilizzate dal CP/M *in modo interno (per esempio i numeri* 27,31). *Il lettore che desidera saperne di più può fare riferimento al manuale di adattamento del* CP/M ("CP/M *guida alle modifiche*").

3.4 I MICROSOFT BASIC

3.4.1 Presentazione generale

Il Microsoft Basic è un Basic molto potente e diffuso. La versione fornita con la Softcard è la versione 5.0. Offre all'utente dell'APPLE delle istruzioni come PRINT USING, CHAIN, WHILE/WEND, l'accesso a cifre in doppia precisione e a manipolazioni di files molto comode. La versione APPLE incorpora le possibilità grafiche e sonore dell'APPLE.
A causa delle dimensioni di memoria necessarie per la grafica ad alta risoluzione, sui dischi forniti si trovano due versioni:

- MBASIC, che è la versione a bassa risoluzione e che occupa 24 K bytes di memoria.
- GBASIC, che è la versione al alta risoluzione e che occupa 30 K bytes di memoria.

Studieremo in questo paragrafo le differenze tra il Microsoft Basic e l'APPLESOFT approfondendo la gestione dei files.

3.4.2 Paragone APPLESOFT-Microsoft Basic

• Possibilità del Microsoft Basic non offerte dall'APPLESOFT

Il Microsoft Basic offre le seguenti possibilità:

- istruzioni supplementari:

CHAIN/COMMON	Passaggio delle variabili ad un altro programma Basic
CALL	Chiamata di subroutines assembler Z80 e 6502
PRINT USING	Uscita dei risultati secondo un determinato formato
PUT #, GET #, FIELD #, WRITE #, INPUT #	
PRINT # PRINT # USING	gestione di files (cf. 3.4.3).
SWAP	Scambia il contenuto di due variabili
WHILE/WEND	Esecuzione di un ciclo finché la condizione è vera
IF..THEN..ELSE	Possibilità di trattamento se la condizione non è verificata
ERASE	Soppressione in memoria della tabella indicata

- Comandi del Basic:

AUTO e RENUM	Generazione dei numeri delle linee e renumerazione delle linee
EDIT	Editor di linee

- Tipi di variabili:
  - Interi, reali a precisione semplice (7 cifre).

• Reali a doppia precisione (16 cifre).
• Nomi delle variabili su 40 caratteri al posto di due.

- Funzioni di stringhe di caratteri:

Assegnamento di MID$, INSTR$, STRING$, HEX$ OCT$

- Operatori logici:

XOR, IMP, EQV, divisione intera, MOD (resto)

- Definizione di funzioni ad argomenti multipli;
- Funzioni grafiche e sonore supplementari:

BUTTON(i)	(i numero manopole di gioco). Consente di sapere se il pulsante della manopola è schiacciato
BEEP(i,j)	Genera un suono di tonalità i e durata j
HSCRN(x,y)	Consente di sapere se il punto (x,y) è acceso sullo schermo
VPOS(o)	Riporta la posizione verticale del cursore.

• Istruzioni utilizzate diversamente in Microsoft Basic e in APPLESOFT

Sono le seguenti:

* FOR... NEXT

Formato APPLESOFT	Si passa sempre almeno una volta nel ciclo. Il test viene fatto alla fine dell'iterazione
Formato MBASIC	Il test viene fatto all'inizio del ciclo. È dunque possibile non entrarci

* INPUT

Formato APPLESOFT	INPUT string; <lista di variabili>
Formato MBASIC	INPUT: string <lista di variabili>: il primo ; sopprime il passaggio alla linea che segue l'INPUT, la virgola sopprime il punto di domanda

* ON ERROR GOTO

È diverso il passaggio del codice dell'errore

Formato APPLESOFT	Passaggio all'indirizzo 222
Formato MBASIC	Uso delle variabili ERR (errore) e ERL (linea)

* RESUME

Formato APPLESOFT	L'esecuzione del programma riprende all'istruzione che ha provocato l'errore
Formato MBASIC	L'esecuzione del programma riprende alla linea specificata nell'istruzione RESUME o a quella che ha provocato l'errore

* TEXT

Formato MBASIC Sbianca lo schermo se si torna dal modo grafico a bassa risoluzione

* GR

Formato APPLESOFT Passaggio in modo grafico a bassa risoluzione con quattro linee di testo, sbiancamento dello schermo e posizionamento del colore a 0

Formato MBASIC Passaggio in modo grafico a bassa risoluzione con o senza quattro linee di testo, riempimento dello schermo col colore desiderato

* HGR, HGR2

Formato APPLESOFT Passaggio in modo grafico ad alta risoluzione con quattro linee di testo

Formato MBASIC Passaggio in modo grafico ad alta risoluzione con scelta del formato dello schermo del colore e dello sbiancamento o meno dello schermo, in pagina 1

* IF THEN ELSE

Aggiunta dell'ELSE

* CALL

Formato APPLESOFT Chiamata senza parametri di S.p. 6502

Formato MBASIC Chiamata di S.p. Z80 col passaggio di parametri 6502 (cfr. par. 3.4.5)

• **Possibilità dell'APPLESOFT non offerte dal Microsoft Basic**

Sono:

DRAW Visualizzazione del disegno sullo schermo a partire da un punto.
XDRAW Stessa cosa col colore inverso.
SCALE Modifica della scala di un disegno.
ROT Rotazione di un disegno.
FLASH Passaggio dello schermo in modo lampeggiante.
SHLOAD Caricamento di un disegno in memoria.

Manca, più che altro, la possibilità di usare dei disegni predefiniti.

3.4.3 Files in MBASIC (o GBASIC)

• **Introduzione**

La gestione dei files è il miglior argomento in favore dell'MBASIC rispetto all'APPLESOFT e al DOS. Possiamo dire che l'MBASIC è più complicato da scrivere come software di base, ma è nettamente più potente, mentre l'APPLESOFT-DOS è molto più facile da scrivere, ma è meno potente. Raccomandiamo vivamente agli utenti che vogliono realizzare delle applicazioni di gestione professionali, di utilizzare la Softcard. In questo modo,

economizzeranno tempo di lavoro e il loro investimento sarà presto ammortizzato.
Nell'MBASIC esistono due tipi di files di dati:

- i files sequenziali;
- i files ad accesso diretto.

La società americana KISS propone, inoltre, una versione dell'interprete MBASIC che consente la gestione di files ad accesso sequenziale indiciato.
Studieremo in questo paragrafo i due tipi di files MBASIC.

• I files ad accesso sequenziale

Le istruzioni e le funzioni offerte sono:

OPEN	"O", numero del file, "nome del file" "I" O output = scrittura di dati I input = lettura di dati
CLOSE	numero del file
INPUT	numero del file, lista di variabili Se dev'essere letta una cifra, MBASIC ignora i blanks, SR, LF e considera l'inizio del numero a partire dal primo carattere diverso dai suddetti e la fine del numero al primo blank seguente Se deve essere letta una stringa, MBASIC ignora gli stessi caratteri elencati per le cifre. Considera il primo carattere come inizio della stringa di caratteri

Si presentano due casi:

- se il carattere è '', sono compresi nella stringa tutti i caratteri fino alla prossima occorrenza di '';
- altrimenti la fine della stringa è contrassegnata da una virgola CR o LF.

WRITE	Scrittura di dati in un file sequenziale. Nel file, i dati sono separati fisicamente da virgole e le stringhe di caratteri sono comprese tra ''
PRINT	Scrittura di dati in un file sequenziale. L'utente dovrà inserire separatori voluti.
PRINT USING	Come l'istruzione PRINT, ma con formattamento dei dati
LINE INPUT	Lettura di una linea per volta
EOF	Funzione che indica se si è arrivati o meno a fine file
LOC	Funzione che indica il numero del prossimo campo da leggere o scrivere

Per scrivere dei dati in un file sequenziale, dovete seguire il seguente iter:

1. OPEN "O", numero file, "nome file"

2. WRITE
   PRINT   dati
   PRINT USING
3. CLOSE,   numero file

Per leggere dei dati dovete fare:

1. OPEN "I",   numero file, "nome file"
2. INPUT
                    dati
   LINE INPUT
3. CLOSE,   numero file

Per aggiungere dei dati in un file sequenziale, non abbiamo il comando equivalente all'APPEND del DOS, ma siccome quest'ultimo non è del tutto affidabile, è preferibile in entrambi i casi copiare il file per aggiungergli in coda i dati.

• I files ad accesso diretto

I files ad accesso diretto occupano meno spazio su disco in quanto i dati sono memorizzati in un codice binario condensato. È possibile accedere ai dati dei records senza dover percorrere tutto il file.
Le istruzioni e funzioni offerte all'utente sono le seguenti:

OPEN "R", numero del file, "nome del file", lunghezza records
CLOSE numero del file
FIELD numero file, larghezza AS stringa...
Quest'istruzione definisce la scomposizione in variabili del buffer di Input/Output. Prima di scrivere nel file, dovete riempire questo buffer, e dopo una lettura nel file, dovete leggerlo
LSET/RSET Queste istruzioni consentono di riempire il buffer di Input/Output mettendo le variabili a sinistra (LSET) o a destra (RSET)

Esempio:
```
10 FIELD 1, 3 AS A$
20 LSET A$="AA" dà AA
                   A$
30 RSET A$="AA" dà AA
                   A$
```

PUT numero file, numero record
Quest'istruzione copia il buffer di Input/Output nel record specificato
GET numero file, numero record
Quest'istruzione copia il record specificato nel buffer di I/O
MKI (intero) Conversione di numeri in stringhe di caratteri codificate in
MKS$ (reale) binario condensato

MKD$ (doppia precisione)	
CVI	
CVS	Funzione inversa
CVD	
LOC	Numero del prossimo record da leggere o scrivere

Eccovi un esempio di un programma in MBASIC che gestisce un file ad accesso diretto contenente un archivio di indirizzi. Potrete paragonare questo esempio con uello del paragrafo 1.6.4.

Esempio. Riprendiamo l'applicazione dell'archivio di indirizzi proposta nel paragrafo 1.4.3 per tradurla in MBASIC aggiungendole la gestione dei buchi (record vuoti) e per evitare di far crescere inutilmente la dimensione del file.
Tutto ciò è ottenuto facilmente grazie all'istruzione FIELD che consente di ridefinire i campi di dati dei records.
La struttura del programma è la seguente:

Linee 120-420 :	- apertura e creazione del file se necessaria - lettura del numero di records - definizione dei campi di dati
Linee 430-580 :	menù e chiamata delle subroutines corrispondenti
Linee 590-630 :	inizio dei sottoprogrammi
Linee 640-1460 :	ricerca di un record: - scelta del metodo di ricerca - lettura del file - test di uguaglianza
Linee 1470-1960:	aggiunta di un record: - acquisizione dei dati - ricerca del primo record libero - scrittura nel file - modifica dei parametri di gestione dei buchi
Linee 1970-2180:	listing del file
Linee 2190-2250:	fine del programma
Linee 2260-2300:	inizio dei sottoprogrammi di secondo livello
Linee 2310-2840:	modifica/soppressione di un elemento - menù - modifica - soppressione
Linee 2850-2990:	apertura del file e definizione dei campi
Linee 3000-3090:	acquisizione del cognome
Linee 3100-3190:	acquisizione del nome
Linee 3200-3290:	acquisizione del tipo di via
Linee 3300-3390:	acquisizione del nome della via
Linee 3400-3460:	acquisizione del numero

Linee 3470–3560: acquisizione della città
Linee 3570–3660: acquisizione del codice postale
Linee 3670–3850: acquisizione del prefisso e del numero di telefono
Linee 3860–3950: visualizzazione del messaggio d'errore "NON PIÙ DI N CARATTERI"
Linee 3960–4060: visualizzazione dei dati

```
10 ' AGENDA DI INDIRIZZI
20 '
30 ' COPYRIGHT B.DE MERLY
40 '
50 ' 11/7/82
60 '
70 ' CREAZIONE DEL FILE ?
80 '
90 HOME:INVERSE
100 PRINT "      AGENDA DI INDIRIZZI  ":PRINT
110 NORMAL
120 INPUT "E' LA CREAZIONE DEL FILE ? (S/N)",A$
130 IF A$<>"N" AND A$<>"S" THEN 120
140 IF A$="N" THEN 320
150 '
160 ' CREAZIONE DEL FILE
170 '
180 PRINT"DURANTE LA CREAZIONE IL FILE PRECEDENTE":PRINT"
    VIENE SOPPRESSO"
190 INPUT "VOLETE CONTINUARE COMUNQUE (S/N)",A$
200 IF A$<>"N" AND A$<>"S" THEN 190
210 IF A$="N" THEN 320
220 GOSUB 2920
230 CLOSE
240 KILL "FICHAD"
250 GOSUB 2930 'APERTURA FILE
260 LSET NB$=MKI$(1)
270 LSET SV$=MKI$(0)
280 LSET PV$=MKI$(2)
290 PUT#1,1
300 CLOSE
310 '
320 ' LETTURA DEL NUMERO DI RECORDS
330 'NEL FILE
340 '
350 GOSUB 2930 'APERTURA FILE
360 GET#1,1
370 NB=CVI(NB$):SV=CVI(SV$):PV=CVI(PV$)
380 CLOSE
390 '
400 'DEFINIZIONE DEL FILE
410 '
```

```
420 GOSUB 2970
430 '
440 '    MENU'
450 '   ******************
460 '
470 HOME:INVERSE
480 PRINT"    *** AGENDA DI INDIRIZZI***":PRINT:PRINT:PRINT
490 PRINT"1- CONSULTAZIONE CON/SENZA MODIFICHE"
500 PRINT"2- ISCRIZIONE (NUOVO INDIRIZZO)"
510 PRINT"3- LISTA DEL FILE"
520 PRINT"4- FINE DEL PROGRAMMA"
530 PRINT
540 INPUT"INSERITE LA VOSTRA SCELTA (1,2,3,4)";A
550 IF A<1 OR A>4 THEN 470
560 NORMAL
570 ON A GOSUB 640,1490,1960,2180
580 GOTO 470
590 '
600 '
610 '     INIZIO DEI SOTTOPROGRAMMI
620 '     ************************
630 '
640 '
650 '***********************************
660 ' RICERCA NEL FILE (1)
670 '***********************************
680 '
690 HOME:INVERSE:PRINT"   *** CONSULTAZIONE ***":NORMAL
700 PRINT:PRINT:PRINT
710 PRINT"DA COSA INIZIA LA RICERCA ?"
720 PRINT
730 PRINT"1- COGNOME"
740 PRINT"2- NOME"
750 PRINT"3- INDIRIZZO"
760 PRINT"4- CITTA'"
770 PRINT"5- CODICE POSTALE"
780 PRINT"6- NUMERO DI TELEFONO"
790 PRINT:INPUT" INSERITE LA VOSTRA SCELTA",A$
800 DIM RECH$(6,2),RECH(6)
810 RECH$(1,1)=SPACE$(15)
820 RECH$(1,2)="COGNOME DA CERCARE"
830 RECH$(2,1)=SPACE$(15)
840 RECH$(2,2)="NOME DA CERCARE"
850 RECH$(3,1)=SPACE$(21)
860 RECH$(3,2)="INDIRIZZO DA CERCARE"
870 RECH$(4,1)=SPACE$(15)
880 RECH$(4,2)="CITTA' DA CERCARE"
890 RECH$(5,1)=SPACE$(5)
900 RECH$(5,2)="CODICE POSTALE DA CERCARE"
```

```
910 RECH$(6,1)=SPACE$(7)
920 RECH$(6,2)="NUMERO DI TELEFONO DA CERCARE"
930 A=LEN(A$)
940 FOR I=1 TO 6
950 RECH(I)=0
960 FOR J=1 TO A
970 IF VAL(MID$(A$,J,1))<>I THEN 1010
980 RECH(I)=1
990 PRINT RECH$(I,2):INPUT" ",B$
1000 LSET RECH$(I,1)=B$
1010 NEXT J
1020 NEXT I
1030 '
1040 ' RICERCA VERA E PROPRIA
1050 ' ************************
1060 '
1070 GOSUB 2920
1080 GET#1,1
1090 NB=CVI(NB$)
1100 IF NB>1 THEN 1120
1110 PRINT"FILE VUOTO":INPUT "BATTERE RETURN PER CONTINUARE",A:RETURN
1120 GOSUB 2960
1130 FOR I=1 TO NB-1
1140 J=J+1 ' NUMERO DI CAMPO
1150 OK=1
1160 GET#1,J
1170 '
1180 ' TEST DI UGUAGLIANZA
1190 '   ******************
1200 '
1210 ' CAMPO LETTO-DATI CERCATI
1220 '   ******************************
1230 '
1240 IF RECH(1)=0 THEN 1260
1250 IF COGN$=RECH$(1,1) THEN OK=OK AND -1 ELSE OK=0
1260 IF RECH(2)=0 THEN 1280
1270 OK = OK AND (NOM$=RECH$(2,1))
1280 IF RECH(3)=0 THEN 1310
1290 A$=NUMERO$+" "+TIPOVIA$+" "+NOMEVIA$
1300 OK = OK AND (A$=RECH$(3,1))
1310 IF RECH(4)=0 THEN 1330
1320 OK = OK AND (CITTA$=RECH$(4,1))
1330 IF RECH(5)=0 THEN 1350
1340 OK = OK AND (CDPOST$=RECH$(5,1))
1350 IF RECH(6)=0 THEN 1370
1360 OK = OK AND (TEL$=RECH$(6,1))
1370 IF OK=0 THEN 1440
1380 HOME:INVERSE:PRINT"POSSIBILI DATI":NORMAL
1390 PRINT:PRINT:PRINT:GOSUB 3960:PRINT
```

```
1400 PRINT"DESIDERATE MODIFICARE O CANCELLARE "
1410 INPUT"QUESTO RECORD (S/N)",A$
1420 IF A$<>"S" AND A$<>"N" THEN 1400
1430 IF A$="S" THEN GOSUB 2310
1440 NEXT I
1450 ERASE RECH$,RECH
1460 RETURN
1470 '
1480 '**********************************
1490 ' AGGIUNTA DI ISCRIZIONI (2)
1500 '**********************************
1510 '
1520 INVERSE
1530 HOME:PRINT"NUOVO INDIRIZZO":PRINT:PRINT
1540 NORMAL
1550 PRINT"SE NON POTETE FORNIRE UNA RISPOSTA"
1560 PRINT" BATTETE RETURN"
1570 PRINT
1580 LSET CODE$="1"
1590 GOSUB 3000 'ACQUISIZIONE DEL COGNOME
1600 GOSUB 3100 'ACQUISIZIONE DEL NOME
1610 GOSUB 3200 'ACQUISIZIONE DEL TIPO DI VIA
1620 GOSUB 3300 'ACQUISIZIONE DEL NOME DELLA VIA
1630 GOSUB 3400 'ACQUISIZIONE DEL NUMERO
1640 GOSUB 3470 'ACQUISIZIONE DELLA CITTA'
1650 GOSUB 3570 'ACQUISIZIONE DEL CODICE POSTALE
1660 GOSUB 3670 'ACQUISIZIONE DEL PREFISSO E DEL NUMERO DI TELEFONO
1670 HOME
1680 INVERSE
1690 PRINT"INFORMAZIONI ACQUISITE"
1700 NORMAL:PRINT:PRINT
1710 GOSUB 3960 'VISUALIZZAZIONE BUFFER
1720 PRINT:INVERSE
1730 INPUT "CONFERMATE QUESTE INFORMAZIONI (S/N) ",A$
1740 IF A$<>"N" AND A$<>"S" THEN 1730 ELSE IF A$="N" THEN 1490
1750 '
1760 ' AGGIUNTA DEL CAMPO NEL FILE
1770 '
1780 PUT#1,PV
1790 GOSUB 2940
1800 IF SV<>0 THEN 1830
1810 ' NON COMPAIONO CAMPI VUOTI NEL FILE
1820 PV=PV+1 :GOTO 1860
1830 PV=SV
1840 GET#1,SV
1850 SV=CVI(SV$)
1860 ' MEMORIZZAZIONE DEI NUOVI DATI
1870 ' NEL PRIMO CAMPO
1880 GET#1,1
```

```
1890 NB=CVI(NB$)+1
1900 LSET NB$=MKI$(NB)
1910 LSET PV$=MKI$(PV)
1920 LSET SV$=MKI$(SV)
1930 PUT#1,1
1940 GOSUB 2980
1950 INPUT A:RETURN
1960 '
1970 '**********************************
1980 ' LISTING DEL FILE (3)
1990 '**********************************
2000 '
2010 GOSUB 2920
2020 GET#1,1
2030 NB=CVI(NB$):PV=CVI(PV$):SV=CVI(SV$)
2040 GOSUB 2960
2050 FOR I=1 TO NB-1
2060 GET#1,I+1
2070 IF CODE$<>"1" THEN 2140
2080 HOME:INVERSE
2090 PRINT"CAMPO NUMERO ";I
2100 NORMAL
2110 PRINT
2120 GOSUB 3960
2130 INPUT "PER CONTINUARE BATTERE RETURN",A
2140 NEXT
2150 HOME:INVERSE:PRINT"FINE FILE ":NORMAL
2160 INPUT "PER CONTINUARE BATTERE RETURN",A
2170 RETURN
2180 '
2190 '**********************************
2200 ' FINE DEL PROGRAMMA (4)
2210 '**********************************
2220 '
2230 CLOSE
2240 NORMAL
2250 END
2260 '
2270 ' INIZIO DEI SOTTOPROGRAMMI
2280 ' DEL SECONDO LIVELLO
2290 '  ***********************
2300 '
2310 '
2320 ' MODIFICA / SOPPRESSIONE
2330 ' ***********************
2340 '
2350 ' DI UN RECORD
2360 ' *****************
2370 '
```

```
2380 PRINT:PRINT"DESIDERATE"
2390 PRINT TAB(20);"MODIFICARE IL CAMPO (1)"
2400 PRINT TAB(20);"CANCELLARE IL CAMPO (2)"
2410 INPUT "INSERITE LA VOSTRA SCELTA ",A
2420 IF A<1 OR A>2 THEN 2380
2430 ON A GOTO 2440,2700
2440 '
2450 ' MODIFICARE UN CAMPO
2460 '  *******************
2470 '
2480 PRINT:PRINT"COSA DESIDERATE MODIFICARE ?"
2490 PRINT:PRINT"1-IL COGNOME":PRINT"2-IL NOME":PRINT"3-L'INDIRIZZO"
     :PRINT"4-LA CITTA'":PRINT"5-
IL CODICE POSTALE":PRINT"6-IL NUMERO
 DI TELEFONO"
2500 PRINT:INPUT"INSERITE LA VOSTRA SCELTA (1,2,3,4,5,6)",A
2510 IF A<>3 THEN 2560
2520 GOSUB 3200  'TIPOVIA
2530 GOSUB 3300  'NOMEVIA
2540 GOSUB 3400  'NUMERO
2550 GOTO 2570
2560 ON A GOSUB 3000,3100,,3470,3570,3670
2570 HOME:INVERSE:PRINT"INFORMAZIONI ACQUISITE":NORMAL
2580 GOSUB 3960
2590 PRINT:PRINT"DESIDERATE"
2600 PRINT"1-MODIFICARE UN ALTRO DATO DEL CAMPO"
2610 PRINT"2-SALVARE LE MODIFICHE EFFETTUATE"
2620 PRINT"3-ABBANDONARE LE MODIFICHE"
2630 INPUT "INSERITE LA VOSTRA SCELTA (1,2,3)",A
2640 ON A GOTO 2440,2650,2690
2650 '
2660 ' REGISTRARE LE MODIFICHE
2670 '
2680 PUT#1,J
2690 RETURN
2700 '
2710 ' CANCELLAZIONE DI UN CAMPO
2720 '  ************************
2730 '
2740 GOSUB 2940 'MODIFICATION FIELD
2750 GET#1,1
2760 PV=CVI(PV$):SV=CVI(SV$)
2770 LSET PV$=MKI$(J)
2780 LSET SV$=MKI$(PV)
2790 PUT#1,1
2800 LSET CODE$="0"
2810 LSET PV$=MKI$(J)
2820 LSET SV$=MKI$(SV)
2830 PUT#1,J
```

```
2840 RETURN
2850 '
2860 ' APERTURA FILE
2870 '  *****************
2880 '
2890 ' DEFINIZIONE DEI CAMPI
2900 '  ***********************
2910 '
2920 CLOSE
2930 OPEN"R",#1,"FICHAD.FIC",101
2940 FIELD#1,1 AS CODE$,2 AS NB$,2 AS PV$,2 AS SV$
2950 RETURN
2960 CLOSE
2970 OPEN"R",#1,"FICHAD.FIC",101
2980 FIELD#1,1 AS CODE$,15 AS COGN$,15 AS NOM$,3 AS TIPOVIA$,
     15 AS NOMEVIA$,3 AS NUMERO$,20 AS C
OMP$,15 AS CITTA$,5 AS CDPOST$,2 AS
INDTEL$,7 AS TEL$
2990 RETURN
3000 '
3010 ' ACQUISIZIONE DEL COGNOME
3020 '  *************
3030 '
3040 INPUT "COGNOME ",A$
3050 N=15:IF LEN(A$)<=N THEN 3080
3060 GOSUB 3870 'VISUALIZZAZIONE DI UN MESSAGGIO DI ERRORE
3070 GOTO 3040
3080 LSET COGN$=A$
3090 RETURN
3100 '
3110 ' ACQUISIZIONE DEL NOME
3120 '  ****************
3130 '
3140 INPUT"NOME",A$
3150 N=15:IF LEN(A$)<=N THEN 3180
3160 GOSUB 3870 'VISUALIZZAZIONE DI UN MESSAGGIOD'ERRORE
3170 GOTO 3140
3180 LSET NOM$=A$
3190 RETURN
3200 '
3210 ' ACQUISIZIONE DEL TIPO DI VIA
3220 '  ***********************
3230 '
3240 INPUT "TIPO DI VIA (VIA,VIALE,PIAZZA..) ",A$
3250 N=3:IF LEN(A$)<=N THEN 3290
3260 GOSUB 3870 'VISUALIZZAZIONE DI UN MESSAGGIO DI ERRORE
3270 GOTO 3250
3280 LSET TIPOVIA$=A$
3290 RETURN
```

```
3300 '
3310 ' ACQUISIZIONE DEL NOME DELLA VIA
3320 '  ***********************
3330 '
3340 INPUT"NOME DELLA VIA ",A$
3350 N=15:IF LEN(A$)<=N THEN 3380
3360 GOSUB 3870 'VISUALIZZAZIONE DI UN MESSAGGIO DI ERRORE
3370 GOTO 3340
3380 LSET NOMEVIA$=A$
3390 RETURN
3400 '
3410 ' ACQUISIZIONE DEL NUMERO
3420 '  ****************
3430 '
3440 INPUT"NUMERO ",A$
3450 LSET NUMERO$=A$
3460 RETURN
3470 '
3480 ' ACQUISIZIONE DELLA CITTA'
3490 '  ******************
3500 '
3510 INPUT"CITTA' ",A$
3520 N=15:IF LEN(A$)<=N THEN 3550
3530 GOSUB 3870 'VISUALIZZAZIONE DI UN MESSAGGIO DI ERRORE
3540 GOTO 3510
3550 LSET CITTA$=A$
3560 RETURN
3570 '
3580 ' ACQUISIZIONE DEL CODICE POSTALE
3590 '  *********************
3600 '
3610 INPUT "CODICE POSTALE ",A$
3620 IF LEN(A$)<6 THEN 3650
3630 PRINT" NON INSERITE PIU' DI 5 CIFRE"
3640 GOTO 3610
3650 LSET CDPOST$=A$
3660 RETURN
3670 '
3680 ' ACQUISIZIONE DEL PREFISSO
3690 '  ***********************************
3700 '
3710 INPUT "PREFISSO TELEFONICO ",A$
3720 IF LEN(A$)<=2 THEN 3750
3730 PRINT"NON INSERITE PIU' DI 2 CIFRE"
3740 GOTO 3710
3750 LSET INDTEL$=A$
3760 '
3770 ' ACQUISIZIONE DEL NUMERO DI TELEFONO
3780 '  ******************************
```

```
3790 '
3800 INPUT"NUMERO DI TELEFONO ",A$
3810 IF LEN(A$)<8 THEN 3840
3820 PRINT"NON INSERITE PIU' DI 7 CIFRE"
3830 GOTO 3800
3840 LSET TEL$=A$
3850 RETURN
3860
3870 '
3880 ' VISUALIZZAZIONE MESSAGGIO D'ERRORE
3890 ' STRINGA TROPPO LUNGA
3900 '  ***************************
3910 '
3920 PRINT:INVERSE
3930 PRINT" NON INSERITE PIU' DI ";N;" CARATTERI "
3940 NORMAL
3950 RETURN
3960 '
3970 ' VISUALIZZAZIONE DEI DATI ACQUISITI
3980 '  ******************************
3990 '
4000 PRINT
4010 PRINT"COGNOME";TAB(20);COGN$
4020 PRINT"NOME";TAB(20);NOM$
4030 PRINT"INDIRIZZO";TAB(20);NUMERO$;" ";TIPOVIA$;" ";TAB(20);NOMEVIA$
4040 PRINT "CITTA'";TAB(20);CDPOST$;" ";CITTA$
4050 PRINT "TELEFONO";TAB(20);"(";INDTEL$;") ";TEL$
4060 RETURN
```

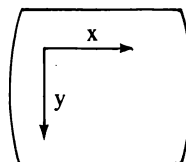
3.4.4 Possibilità grafiche e sonore

Il Microsoft Basic standard è stato esteso sull'APPLE II per consentire al possessore di una scheda Z80 di conservare le possibilità grafiche del monitor dell'APPLESOFT. Le funzioni offerte sono le seguenti:

POS	Funzione che indica la posizione orizzontale del cursore sullo schermo
VPOS	Funzione che indica la posizione verticale del cursore sullo schermo
BUTTON (numero manopola)	Funzione che indica se il pulsante della manopola è schiacciato (valore —1) o no (valore 0)
BEEP I,J	Emissione di un suono di tonalità I e durata J (I compreso tra 0, che è la tonalità più elevata, e 255, che è la tonalità più bassa; J compreso tra 0 e 255 i.e. 1)

GR numero schermo, numero colore
: Passaggio in modo grafico a bassa risoluzione sullo schermo 0–1 (4 o 0 linee di testo) con il colore desiderato

COLOR = numero di colore (da 0 a 15) sullo schermo

PLOT x,y
: (x compreso tra 0 e 39, y tra 0 e 47)

VLIN y1, y2 AT x:	Tracciato di una linea verticale dal punto (x,y1) al punto (x,y2) in modo grafico a bassa risoluzione
HLIN x1,x2 AT y:	Tracciato di una linea orizzontale dal punto (x1,y) al punto (x2,y) in modo grafico a bassa risoluzione
SCRN (x,y)	Funzione che ritorna il valore del colore del punto (x,y)
TEXT	Torna in modo testo con lo sbiancamento dello schermo, se torna dal modo a bassa risoluzione, senza sbiancamento se torna dal modo ad alta risoluzione; in questo secondo caso si posiziona alla linea 24
HTAB numero colonna	Spostamento del cursore verso la colonna specificata modulo 40
VTAB numero linea	Spostamento del cursore verso la riga specificata modulo 24
INVERSE	Passaggio dello schermo in modo inverso (caratteri neri su fondo bianco)
NORMAL	Ritorno dello schermo in modo normale (caratteri bianchi su fondo nero)
PDL (numero manopola)	Lettura del valore specificato dalla manopola di gioco (0–255)

*Come per il monitor dell'*APPLE II, *tra due chiamate di funzione, dovete aspettare. A questo scopo inserite la linea:*

FOR Z = 1 TO 10:NEXT Z

Possibilità grafiche ad alta risoluzione offerte dal GBASIC:

HGR <numero schermo>, <numero colore>

dove numero schermo è compreso tra 0 e 3
numero colore è compreso tra 0 e 12.

Inizializzazione del modo grafico ad alta risoluzione.
I numeri da 0 a 3 dello schermo corrispondono ai seguenti formati:

Numero schermo	Sbiancamento schermo	Modo schermo
0	SI	280×160, 4 linee di testo
1	SI	280×192, senza testo
2	uguale allo 0 senza sbiancamento dello schermo	
3	uguale all'1 senza sbiancamento dello schermo	

HCOLOR = <numero colore>

Posizionamento del colore desiderato (0-12):

0 nero 0	4 nero 1	8 nero 2	da usare con il verde e il viola
1 verde	5 arancione	9 bianco 2	
2 viola	6 blu	10 nero 3	
3 bianco 0	7 bianco 1	11 bianco 3	con l'arancione e il blu
		12 negativo	

I diversi colori neri o bianchi corrispondono a tratti di diversa larghezza.

HPLOT <x1,y1> TO <x2,y2>---

visualizzazione di un punto o tracciato di una linea in modo grafico ad alta risoluzione.
HSCRN (x,y) indica se esiste sullo schermo un punto di coordinate (x,y) (-1 se non c'è, 0 altrimenti).
Siccome queste funzioni sono simili nell'uso a quelle del monitor e dell'APPLESOFT, il lettore interessato potrà rifarsi al capitolo 3.
A nostro parere, per programmare dei giochi, è preferibile poter disporre di forme grafiche preregistrate (cf. volume 1) e semplici da spostare.
La presenza di funzioni grafiche nel Microsoft Basic può permettere una migliore presentazione dei risultati sullo schermo. Per questo motivo è utile conoscerle.

3.4.5 Chiamata di subroutines assembler e indirizzi di memoria

Esistono diverse possibilità per chiamare subroutines assembler dall'MBASIC:

- funzione monoargomento USR equivalente alle funzioni FN del Basic;
- chiamata di sottoprogrammi assembler Z80 mediante l'istruzione CALL;
- chiamata di sottoprogrammi assembler 6502 mediante l'istruzione CALL%.

• Funzioni monoargomento USR

L'istruzione

DEF USR<digit> = <espressione intera>
0–9

permette di definire 10 funzioni assembler 8080 specificando il loro indirizzo

Esempio: DEF USR3 = 2048

La chiamata di diverse funzioni viene così effettuata:

USR<digit>(argomento)
0–9

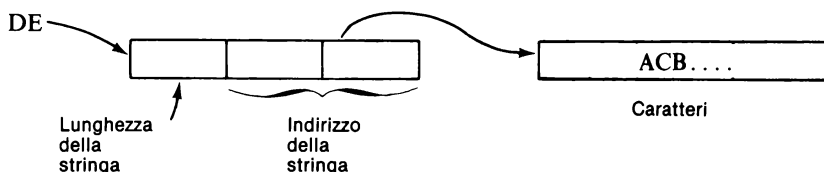
Esempio: B = USR4(2000)

Al momento della chiamata di una funzione USR, il registro A (accumulatore) indica il tipo di argomento. Il valore di A può essere:

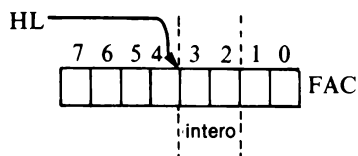
2 intero (su 2 bytes)
3 stringa di caratteri
4 numero reale a precisione semplice
8 numero reale a doppia precisione

Al momento del ritorno dalla funzione, l'accumulatore deve contenere il tipo del parametro di ritorno.
Se il parametro di chiamata è una stringa di caratteri, i registri DE puntano a 3 bytes chiamati descrittori di stringa

DE		ACB...
lunghezza della stringa	indirizzo della stringa	caratteri

Il seguente esempio inverte i primi due caratteri di una stringa di caratteri:

```
10 POKE &H8000,&HEB      'XCHG
20 POKE &H8001,&H23      'INX     H
30 POKE &H8002,&H5E      'MOV     E,M
40 POKE &H8003,&H23      'INX     H
50 POKE &H8004,&H56      'MOV     D,M
60 POKE &H8005,&H2B      'DEX     H
70 POKE &H8006,&H2B      'DEX     H
```

```
80 POKE &H8007,&HEB         'XCHG
90 POKE &H8008,&H7E         'MOV     A,M
100 POKE &H8009,&H23        'INX     H
110 POKE &H800A,&H46        'MOV     B,M
120 POKE &H800B,&H77        'MOV     M,A
130 POKE &H800C,&H2B        'DEX     H
140 POKE &H800D,&H70        'MOV     M,B
150 POKE &H800E,&H3E        'MVI     A,3H
160 POKE &H800F,&H3
170 POKE &H8010,&HC9        'RET
180 DEF USR=&H8000
190 INPUT A$
200 A$=USR(A$)
210 PRINT A$
220 GOTO 190
```

Se il parametro di chiamata è un intero su 2 bytes, i registri HL puntano ad una parte di un buffer chiamato "Floating point accumulator" da 8 bytes.

FAC3 = bit meno significativo (resto della divisione per 256)
FAC2 = bit più significativo (quoziente della divisione per 256)

Il seguente esempio mostra un esempio di divisione per 256 di un intero:

```
10 POKE &H8000,&HE5        'PUSH  H
20 POKE &H8001,&H23        'INX   H
30 POKE &H8002,&HAF        'XRA   A
40 POKE &H8003,&H77        'MOV   M,A
50 POKE &H8004,&HE1        'POP   H
60 POKE &H8005,&H3E        'MVI   A,2H
70 POKE &H8006,&H2
80 POKE &H8007,&HC9        'RET
90 DEF USR=&H8000
100 INPUT A%
110 A%=USR(A%)
120 PRINT A%
130 GOTO 100
140 END
```

Per un numero reale, HL punta a FAC3 con

FAC3	mantissa	bit — significativo
FAC2		
FAC1		bit + significativo
FAC0	esponente	

Per un numero reale a doppia precisione, i valori da FAC7 a FAC4 rappresentano 4 bytes di mantissa in più. HL punta a FAC7.
Il parametro di ritorno di USR deve essere dello stesso formato di un parametro dello stesso tipo alla chiamata.

• Sottoprogrammi assembler Z80

Rispetto alle funzioni USR, consentono l'utilizzo di più parametri.
L'istruzione che permette di chiamare una subroutine è:

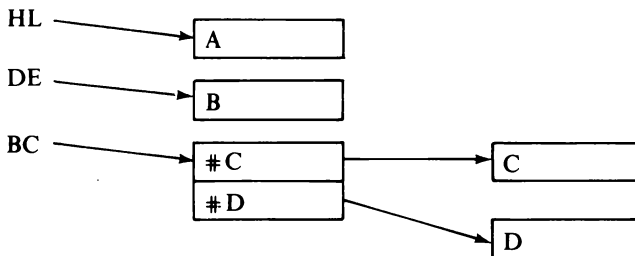
CALL<nome della variabile> <lista di argomenti>

La variabile specificata contiene l'indirizzo al quale verrà passato il controllo.
Il sottoprogramma chiamato deve essere a conoscenza del numero di parametri. Il loro passaggio è realizzato nel seguente modo:

HL indirizzo dell'argomento numero 1
DE indirizzo dell'argomento numero 2
BC indirizzo di una zona di memoria contenente gli indirizzi degli altri parametri

Per esempio, se l'MBASIC incontra l'istruzione:

CALL SPROG(A,B,C,D)

crea il seguente contesto:

È compito del sottoprogramma assembler recuperare i parametri che gli vengono passati dal MBASIC.

• **Sottoprogrammi assembler 6502**

Il formato dell'istruzione di chiamata di subroutines 6502 è il seguente:

CALL% nome s.p. (lista di parametri)

Tra Basic e assembler possono essere passati al massimo tre parametri. Questi vengono trasmessi rispettivamente nei registri:

6502 A(1)
X(2)
Y(3)

• **Funzione VARPTR**

Consente di conoscere l'indirizzo di una variabile in memoria. Il suo formato è il seguente:

PT X = VARPTR(X)

dove X è una variabile di qualunque tipo inizializzata.
Questa funzione può essere utile per chiamare dei sottoprogrammi assembler Z80 o 6502.

3.5 LINGUAGGI DISPONIBILI

3.5.1 Introduzione

Numerosi linguaggi e software di base sono adattabili molto facilmente al CP/M dell'APPLE II tramite una scheda di interfaccia V24, e ad un programma di trasferimento adeguato. La ditta Microsoft ha così trasferito i suoi linguaggi sulla Softcard e, in questo modo, potete disporre del Fortran, del Cobol, di un compilatore Basic, del Pascal, del Lisp, o di programmi evoluti per il trattamento dei testi, come il Wordstar.

3.5.2 Sistema di sviluppo assembler ALDS

Il prodotto A.L.D.S., sviluppato specialmente dalla Microsoft per l'APPLE II, vi consentirà di programmare nei tre linguaggi assembler dei microprocessori Zilog Z80, Intel 8080 e Rockwell 6502. Inoltre, il sistema A.L.D.S. è il solo assemblatore (oltre a quello della scheda dei linguaggi) che permette l'uso di macroistruzioni, l'assemblaggio condizionale di diverse zone di programmi e che genera codice binario rilocabile (vale a dire inseribile dovunque in memoria dopo il collegamento, cioè dopo aver generato dei "LINK").
Il sistema A.L.D.S. contiene i seguenti software:

a) M80: macroassembler del sistema;
b) L80: collegatore che consente di riunire programmi assembler, Basic

compilato, Fortran, Cobol, ... e che genera un programma caricabile in memoria;
c) CREF80: generatore di riferimenti incrociati (vale a dire che forniscono la lista delle variabili e il loro uso);
d) DDT65: programma di aiuto alla messa a punto equivalente al software DDT del CP/M, per il microprocessore 6502 (analisi e modifica dei registri, della memoria; decodifica di zone di memoria; ecc.),
e) CPMXFER: software per il trasferimento di files binari contenenti dei programmi 6502 pronti per il caricamento, dal sistema CP/M al DOS APPLE.

3.5.3 Compilatore Basic

Accrescere la velocità di esecuzione di un programma non è l'unico interesse del compilatore Basic di Microsoft. A seconda dei programmi compilati, la velocità di esecuzione è in media da 10 a 20 volte più veloce e può diventare fino a 30 volte superiore per dei software che usano al massimo variabili intere.
Inoltre, potrete raggruppare dei programmi Basic con Fortran o Cobol. Il codice generato dal compilatore è altamente ottimizzato. Praticamente, può essere interpretabile solo da un professionista e in questo modo, facilita la protezione dei vostri programmi.
Occorre notare tuttavia, che un certo numero di istruzioni del Basic interpretato non sono utilizzabili in Basic compilato.

3.5.4 Il compilatore Fortran

Il Fortran Microsoft è conforme alla norma americana ANSI 1966, tranne l'assenza dei numeri complessi. Il linguaggio Fortran, creato specialmente per risolvere i problemi numerici, è diventato il linguaggio più utilizzato per le applicazioni scientifiche. Esistono numerose biblioteche di sottoprogrammi Fortran 66 facilmente adattabili all'APPLE II.
Siccome il Fortran è già utilizzabile con la scheda dei linguaggi, non lo sviluppiamo.
*Il software di controllo degli I/O non standard dell'*APPLE II *è facilmente adattabile al Fortran.*

3.5.5 Il compilatore Cobol

Da molti anni, il linguaggio Cobol domina le applicazioni di gestione informatiche. Esso è adatto, in particolar modo, alla manipolazione e alla gestione di grandi quantitativi di dati. Questi vengono ordinati nelle strutture logiche, in maniera gerarchica.
Il Cobol Microsoft è stato sviluppato specialmente per un ambiente di microcalcolatori con un comportamento interattivo, un formattamento di

schermo e una messa a punto dei programmi interattiva. Ammette dei numeri rappresentati su 18 bits, necessari alla specificazione delle variabili e realizza le operazioni interne su 38 bits.
Permette la gestione di 4 tipi di files. Assieme al Cobol viene fornito il software di selezionamento "Microsoft Sort", il quale consente di riorganizzare dei files la cui dimensione arriva fino a 2 miliardi di bytes.

3.5.6 Il LISP

Il LISP è un linguaggio ad alto livello che consente la manipolazione di simboli e il trattamento di espressioni simboliche. È stato creato da e per ricercatori di intelligenza artificiale, ma viene utilizzato anche da scienziati e matematici.
Le strutture e gli oggetti del LISP Microsoft sono adatti a questi usi, grazie alla sua natura ricorsiva e alla sua capacità di trattare liste di dati. I programmi e i dati sono rappresentati da liste concatenate. Questo rende più facile la riorganizzazione dei dati per le diverse applicazioni e elimina il bisogno di allocare delle zone di memoria alle diverse parti del programma.
Facili da imparare e da usare, gli oggetti del LISP permettono di sviluppare delle espressioni complete. Con le 24 funzioni di base del LISP Microsoft, potrete creare il vostro linguaggio.
Il muSTAR fornito con il muLISP è un text editor che facilita la messa a punto dei programmi.

3.5.7 Sistema muSIMP/muMATH

Questo sistema propone la manipolazione di simboli matematici per fare l'algebra, i calcoli trigonometrici, le derivazioni formali, le integrazioni formali (per esempio Xdx = XX/2).
È diviso in due parti chiamate muMATH e muSIMP (linguaggio nel quale è scritto muMATH).
muMATH è un insieme di programmi che procurano delle facilitazioni nel trattamento algebrico e analitico. Le espressioni sono semplificate man mano che vengono prese. Potete creare il vostro software, creare delle lezioni secondo i vostri desideri, ecc.
muMATH è scritto nel linguaggio muSIMP, sottoinsieme del LISP, concepito specialmente per implementare funzioni algebriche. Serve anche a sviluppare delle applicazioni di intelligenza artificiale.

3.5.8 Pascal

Il Pascal è già utilizzabile con la Softcard. Siccome la scheda dei linguaggi e il Pascal UCSD sono largamente diffusi, non ne parleremo.

CAPITOLO 4

Schede di espansione per l'APPLE II

4.1 INTRODUZIONE

Per l'APPLE II, che è il personal più ricco di espansioni, sono disponibili numerose schede di espansione. Queste schede si dividono in 4 categorie principali:

- Schede che consentono il collegamento di periferiche "standard".
- Schede di espansione della memoria.
- Schede che consentono di disporre di un software più completo.
- Schede che consentono l'uso di un altro microprocessore e in generale di un altro sistema operativo.

Questo capitolo presenta senza entrare nei dettagli, le schede principali.

4.1.1 Schede di collegamento di periferiche standard

Si tratta principalmente di schede di interfaccia che consentono il collegamento di periferiche, non concepite esclusivamente per l'APPLE II (dischi duri, terminali, stampanti, modem...).
Tranne i dischi che necessitano di un'interfaccia speciale, le periferiche classiche sono collegate grazie a tre principali tipi di schede:

- schede di interfaccia parallela, gli 8 bits di un byte sono trasmessi in parallelo;
- schede di interfaccia seriale asincrona che trasmettono successivamente gli 8 bits di un byte su uno stesso filo senza che gli altri componenti (APPLE, periferica) siano stati precedentemente sincronizzati;
- schede di interfaccia seriale sincrona per la quale gli altri componenti sono permanentemente sincronizzati tra loro mediante un orologio.

L'APPLE propone le seguenti tre schede di interfaccia:

- interfaccia di comunicazione funzionante in anello di corrente 20 mA alla velocità di 110 o 300 bits/secondo, adattata ai terminali schermo-tastiere e ai modem acustici;
- interfaccia seriale RS232 (da A a E) asincrona funzionante ad una velocità compresa nell'intervallo 75-19000 bits/secondo e uso più semplice;
- interfaccia parallela per stampanti.

Citiamo qui sotto le schede di interfaccia della società California Computer System in quanto sono delle ottime schede:

CCS 7712: scheda di interfaccia seriale sincrona
CCS 7710: scheda di interfaccia seriale asincrona
CCS 7720: scheda di interfaccia parallela.

La scheda EPSON, presentata nel volume 1 al paragrafo 2.3, è compresa in questa categoria.

4.1.2 Schede di espansione della memoria

Citiamo tra queste schede:

- La RAMCARD, di origine Microsoft, (capacità di 16 K bytes, questo costituisce un buon complemento alla Softcard).
- La scheda di SATURN SYSTEM che permette espansioni di 16, 32, 64 o 128 K bytes. Questa scheda viene fornita con un complemento al software che consente di metterla in funzione. Nella versione da 128 K bytes, essa permette di simulare files in memoria centrale; questo riduce considerevolmente i tempi di accesso. Un software complementare permette di utilizzare VISICALC usando il supplemento di memoria disponibile; questo riduce il numero di accessi a disco e migliora le prestazioni durante l'esecuzione.
- La scheda LEGEND porta 128 K bytes di memoria RAM, suddivisa in 8 blocchi da 16 K bytes. Essa può anche essere consegnata con un complemento al software compatibile con il DOS 3.3; questo consente di simulare i files su disco.

In ogni caso non bisogna dimenticare, al termine del lavoro, di copiare su disco i files modificati per non perderli.

4.1.3 Schede che consentono di disporre di un software di base più completo

Citiamo le due più conosciute:

- La scheda dei linguaggi, che consente in particolare di disporre del Pascal UCSD e che è stata oggetto di un capitolo speciale, tenuto conto della sua importanza.
- La scheda M/DOS, che è stata sviluppata dalla società francese MIS, è

essenzialmente destinata ad applicazioni di gestione. La scheda M/DOS è un importante complemento al DOS 3.3, in particolar modo per la gestione dei files (files sequenziali indiciati a una o più chiavi), per l'acquisizione e l'uso di editor (uso molto semplice delle maschere di acquisizione e di memorizzazione), possibilità di reti, ecc.

4.1.4 Schede che consentono l'uso di un altro processore

La scheda più conosciuta è la Softcard che permette di disporre del sistema operativo CP/M. Comunque esistono altre schede meno note:

- Schede 8088, che consentono di simulare il Personal Computer dell'IBM. Queste schede hanno un microprocessore 8088 e 64 o 128 K bytes.
- Scheda 6809, che consente di disporre del sistema operativo OS9 e di una maggiore rapidità di esecuzione in Basic, Fortran e Pascal. Questa scheda porta un 6809 alla frequenza di 1 MHz mentre la sua velocità nominale è di 2 MHz.

4.1.5 Schede che consentono applicazioni di misure

Queste schede sono molto numerose, citiamo:

- La scheda A/D e D/A, messa in commercio dall'APPLE per l'acquisizione di dati analogici e l'uscita di informazioni sotto forma analogica.
- La scheda ADALAB, messa in commercio della ALPHA SYSTEMES, consente non solo gli I/O analogici, ma anche gli I/O su 8 o 16 bits in parallelo e un orologio per segnare l'acquisizione. Questa scheda è fornita con un po' di software in modo da facilitare l'apprendistato dell'utente.

4.1.6 Schede da 80 colonne

La scheda Videotherm permette la visualizzazione di 24 linee da 80 caratteri di dimensioni 7×9 punti in lettere maiuscole e minuscole. Il gioco di caratteri è programmabile mediante software nelle dimensioni da 8×2 a 16 punti. È possibile avere dei giochi di caratteri inglesi, francesi, tedeschi, giapponesi, ecc. Il creatore della scheda Videotherm propone, inoltre, una scheda di riconfigurazione di tastiera, e una scheda di selezione dei modi video 40 o 80 colonne.

Fornitore: VIDEX
897 N.W. Grant Ave,
Corvallis
Oregon 97330
USA

Altre schede da 80 colonne: la Sup'R terminal compatibile con la scheda M/DOS descritta al paragrafo 4.2.

4.1.7 Schede da digitalizzare

Queste schede trasformano un'informazione del modo esterno (analogico) in un'informazione comprensibile all'APPLE II. Oltre alle schede CA/D citate nel paragrafo precedente, che lavorano su una dimensione, possiamo citare dei sistemi a due dimensioni (tabellina grafica, scheda Digisector...) o a tre dimensioni.
Ritorneremo su queste schede nel paragrafo 4.7.

4.1.8 Sintesi e riconoscimento sonoro

Tra le schede che consentono di manipolare i suoni sull'APPLE II:

SUPERTALKER	: digitalizzatore e sintetizzatore della voce umana (2 min per disco)
MUSIC SYSTEM	: sintetizzatore di musica con composizione con l'aiuto delle manopole, della tastiera o di uno stiletto magnetico
VOICE INPUT MODULE	: riconoscimento di un vocabolario limitato, registrato precedentemente
VOICE BOX	: sintesi di un testo digitato alla tastiera con l'aiuto di un dizionario di fonemi registrati su disco.

4.2 SCHEDA M-DOS (CHIAMATE ORA MEM/DOS)

Questa scheda è destinata soprattutto alle applicazioni di gestione:

- gestione dei files da archiviare;
- gestione dello schermo (maschera di acquisizione);
- gestione dell'editor (maschera di editing).

4.2.1 Gestione dei files

Questa scheda consente l'uso dei files sequenziali, indiciati, multichiave, e la gestione dei files relativi e dei files sequenziali classici. Qualunque sia il tipo del file, la gestione dei dischi è dinamica. La dimensione del file non deve superare 60.000 campi. I dischi gestiti possono essere dei dischi da 5 pollici, 8 pollici o dei dischi rigidi.

4.2.2 Gestione dello schermo

Per la gestione dello schermo con delle maschere. Al momento della scelta, il controllo è effettuato dal software contenuto nella scheda.
La scelta è estremamente rapida, con un eccellente controllo di verosimiglianza dei dati. La stessa maschera può essere utilizzata da più programmi.

4.2.3 Gestione dell'editing

Gli editing, in particolare modo quelli verso la stampante, possono essere effettuati mediante delle maschere; questo è ciò che dà l'accesso all'istruzione PRINT USING e che normalmente non è disponibile con il Basic dell'APPLE.
Questa scheda M/DOS 6502 contiene essenzialmente dei programmi scritti in memoria EPROM; in questo modo libera completamente la memoria RAM dell'APPLE.
Esiste una versione monoutente e una versione multiutente. Nella versione multiutente, essa consente a più APPLE di proteggere delle periferiche (memoria di massa e stampante) quando dei dischi sono in comune:

- spartizione dei files;
- accesso simultaneo ad uno stesso file da parte di più utenti con protezione di campi. Per un utente è possibile anche riservare dei programmi e dei files memorizzati su un disco comune.

Fornitore: MICRO INFORMATIQUE SERVICE
3, Rue Meyerber
06000 NICE
Tel. (93) 877467

4.3 SCHEDE MEM/PLOT

Questa scheda, fornita dalla MIS, serve alle applicazioni grafiche. Contiene 20 K bytes di memoria EPROM che gestiscono da 1 a 3 periferiche grafiche, come:

- matita ottica;
- tavola per tracciare da 1 a 16 colori;
- stampante grafica;
- tabellina di digitalizzazione;
- schermo grafico.

In generale, per le applicazioni grafiche, la configurazione del sistema è composta da uno schermo grafico, da una tavola per tracciare i colori e da una tabellina di digitalizzazione.
Il modo di operare consiste nell'aggiungere, al Basic dell'APPLE, un modulo grafico, che è un sottoinsieme molto simile del modulo grafico della futura norma Basic.
Inoltre, questo sistema traccia gli assi - e la scala sugli assi - mediante istruzioni del tipo GR"XAXIS", per l'asse delle ascisse, e GR"YAXIS", per l'asse delle ordinate. Come nel modulo grafico, questa scheda consente la gestione di subroutines grafiche (definizione, tracciato, trasformazione dell'immagine: rotazione, translazione, effetto zoom).

4.4 SCHEDE CHE CONSENTONO L'USO DI UN ALTRO PROCESSORE

4.4.1 Schede 6809

Questa scheda contiene un 6809 funzionante a 1 MHz. Il 6809 è più rapido del 6502 in quanto dispone di registri da 16 bits. La scheda dispone del sistema operativo OS/9 e del Basic sotto OS/9 che accetta delle variabili locali, dei sottoprogrammi con passaggio di parametri, e delle funzioni matematiche più numerose di quelle del Basic dell'APPLE (ad esempio ARCSIN e ARCCOS).
Questa scheda esegue i programmi scritti in Pascal o in Fortran, con le migliori prestazioni.

4.4.2 Schede Z80

Abbiamo già trattato la Softcard. Esistono delle schede concorrenti:

- La CP/M Card, che consente di disporre del CP/M versione 3, a condizione di aggiungervi la scheda da 80 colonne.
- L'APPLI-Card, che contiene uno Z80 64 K bytes di memoria RAM, la gestione dello schermo con 70 colonne e i softwares CP/M, SBBO, WORDSTAR. Una versione di questa scheda funziona a 6 MHz (migliori prestazioni durante l'esecuzione).
- La Z-Card.
- La CP/M-Card, più recente, che è consegnata con il CP/M+ (versione 3), il C-Basic e qualche altro programma. Questa scheda contiene uno Z80 B a 6 MHz e 64 K di memoria RAM.

Fornitore: APPLI-CARD
PCPI
16776 Bernardo Center Drive
San Diego, Cal 92128
USA

4.4.3 Schede 8088

Diverse schede 8088 contengono dei complementi alla memoria RAM (in genere, da 24 a 128 K) per poter utilizzare il sistema operativo MS-DOS, disponibile sul personal computer dell'IBM.
In pratica, queste schede presentano dei vantaggi solo per le configurazioni che dispongono di almeno due unità disco e di una stampante.

4.5 SCHEDE PER APPLICAZIONI NEI LABORATORI

Queste schede sono molto numerose; non possiamo darne che un'idea.
Esistono delle schede di conversione analogica, numerica, delle schede per

Input/Output TTL, delle schede che consentono l'interfaccia con il BUS IEEE-488 molto usato nelle applicazioni di acquisizione di misure in laboratorio (HEWLETT PACKARD è all'origine di questo bus). Una scheda originale è la 85aSCOPE che trasforma l'APPLE in oscilloscopio. Essa permette, con un APPLE II, di acquisire misure su una frequenza che può arrivare fino a 50 MHz. La conversione analogica digitale ha una precisione di 8 bits.

Fornitore: NORTHWEST INSTRUMENT SYSTEMS
P.O. Box 1309
Beaverton, Oregon 97075 Tel. (503) 297143 - USA

4.6 SCHEDA P.A.R. (SCHEDA PROCESSORI ARITMETICI RAPIDI)

Si tratta di una scheda particolare per effettuare dei calcoli rapidi in virgola mobile. Un'opzione "memoria morta" di questa scheda permette di calcolare la trasformata di FOURIER rapida. Un'altra scheda consente di effettuare operazioni su 16 e 32 bits con prestazioni migliori di quelle ottenute con l'APPLESOFT. Questa scheda permette anche il calcolo di funzioni matematiche del tipo SENO, LOGARITMO, ecc...

Fornitori: 1) IEF 193, Rue de Javel
75015 PARIS
Tel. (1) 8280601
2) Scheda CCS 7811B (circuito AMD9511)
California Computer Systems

4.7 SCHEDE DA DIGITALIZZARE

4.7.1 Tabellina grafica

La tabellina grafica APPLE è un sistema a due dimensioni sul quale realizzate dei disegni con l'aiuto di uno stiletto magnetico. IL disegno tracciato viene, a questo punto, digitalizzato per poter essere trattato dal sistema APPLE II e visualizzato sullo schermo.
Questo permette, ad esempio, di fare creazioni artistiche sull'APPLE II.

Fornitore: APPLE SEEDRIN
Av. de l'Oceanie
Z.A. de Courtaboeuf
91... Les Ulis

4.7.2 Schede Vision

Si tratta di un insieme contenente: telecamera specializzata che dà un'immagine di definizione 128×256 (inferiore allo schermo dell'APPLE II), un

supporto, la scheda di espansione, il cavo e il software di messa in funzione. Questo materiale esiste nelle versioni APPLE II, IBM-PC, TRS80, Commodore o Sinclair ZX81.

Fornitore: MICRON TECHNOLOGY INC.
2805 East Columbia Road
Boise, Idaho 83706

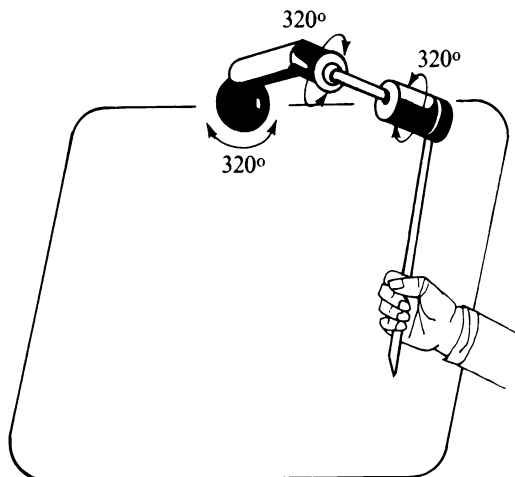
Un'altra scheda, la Digisector, offre una risoluzione dello stesso ordine con trattamento possibile ad ogni secondo.
Queste schede sono utili nelle applicazioni di riconoscimento delle forme riprendendo ad esempio l'immagine digitalizzata nella pagina grafica ad alta risoluzione dell'APPLE II.

4.7.3 Scheda da digitalizzare in tre dimensioni

Sistema composto da una tabellina grafica a due dimensioni sulla quale è stato adattato un braccio articolato. Questo permette di seguire il contorno di un oggetto, che viene riprodotto in prospettiva sullo schermo. Il braccio è composto da tre segmenti che possono ruotare di 320 gradi, e la tabellina ha una superficie di 13,5 × 16 pollici.

44,5 × 52,8 centimetri

Il modello standard è destinato all'APPLE II. Esiste un modello, detto "professionale", collegabile all'APPLE II e al personal computer IBM-PC, che contiene un quarto asse di rotazione.

Fornitore: MICRO CONTROL SYSTEM INC.
143 Tunnel Road
Vernon, Connecticut 06066 - USA

4.8 SCHEDE PER IL TRATTAMENTO DELLA PAROLA O DELLA MUSICA

4.8.1 Trattamento mediante campionatura. Scheda Super Talker

Questa scheda permette di selezionare la voce umana mediante un microfono, di trattarla (programmi di confronto o di riconoscimento vocale sono facilmente realizzabili) e di restituirla tramite un altoparlante o un amplificatore Hifi.
L'insieme Super-Talker contiene una scheda di espansione, un microfono, un altoparlante e i softwares di campionatura e di restituzione del vocabolo, di memorizzazione su disco della parola digitalizzata.
Le quattro velocità di campionatura: sono utilizzabili 500 bytes/secondo, 1000 b/s o 4000 b/s che corrispondono rispettivamente alle frequenze 4000 Hz, 8000 Hz, 16000 Hz o 32000 Hz. Alla velocità di 2 K bytes/secondo (16 KHz), è possibile memorizzare su un disco 50 s di parola.
Questa scheda può essere utilizzata per suoni diversi dalla voce umana a condizione che la frequenza sia nell'intervallo 300–3000 Hz (intervallo di frequenza della voce).

Fornitore: MOUNTAIN HARDWARE INC
300 Harvey West Blvd
Santa Cruz, CA95060
USA

4.8.2 Scheda "Music System"

Questa scheda musicale mette la musica a portata di tutti. Offre un sintetizzatore digitale a 16 note con uscita in stereofonia. Potete così, con l'aiuto di una matita luminosa, comporre le vostre partiture...

Fornitore: MOUNTAIN HARDWARE INC
300 Harvey West Blvd
Santa Cruz, CA95060
USA

4.8.3 Scheda "Voice Input". Modulo di riconoscimento della parola

Questa scheda consente di riconoscere parole preregistrate. Infatti, il riconoscimento è possibile solo su parole isolate.

Esempio: scheda VMC 2020.

Sulla scheda sono presenti 8 K bytes di memoria RAM e 4 K bytes di memoria ROM. La sua capacità di riconoscimento è limitata a 80 parole; il

tempo necessario per il riconoscimento è inferiore a 150 ms; questo impone un intervallo minimo di 160 ms tra due parole successive. La scheda viene consegnata con un microfono e la sua interfaccia.

Fornitore: VOICE MACHINE COMMUNICATIONS INC
10522 Covington Circle
Villa Park
California 92667
USA

4.8.4 Scheda Voice Box

Questo sistema consente di sintetizzare un testo preso da tastiera scomponendolo in fonemi (suoni di base - bi, ti, ...) e sintetizzandoli. In ROM, sulla scheda di espansione, si trova un dizionario di 64 fonemi adattato alla lingua inglese. Il passaggio ad un altro linguaggio si effettua cambiando la ROM; finora esiste solo la versione inglese.

Fornitore: THE ALIEN GROUP
Department PT-2
27W 23 St
New York 10010
USA

Indice analitico

GUIDA PER L'APPLE

L'APPLE STANDARD

Benoît de Merly

Questo libro si rivolge a tutti coloro che utilizzano un computer Apple e si rivelerà presto indispensabile nella ricerca delle migliori possibilità di questo personal.
Ciascuno vi troverà l'informazione pratica di cui ha bisogno qualunque sia l'applicazione prevista: gestione, calcoli scientifici, giochi, grafica, acquisizione di dati, controllo di processo. Un'opera completa, chiara e pratica che dovrebbe diventare familiare a tutti gli utenti Apple.

L'autore, un ingegnere civile laureato in informatica, ebbe l'idea di scrivere questo libro mentre utilizzava lui stesso un Apple: si è perciò posto le domande che l'utente è portato a porsi se vuol ottenere dal suo computer il massimo delle prestazioni e padroneggiarlo perfettamente.

GUIDA PER L'APPLE

VOLUME 1	VOLUME 2	VOLUME 3
L'APPLE STANDARD	LE ESTENSIONI	LE APPLICAZIONI

I.S.B.N. 88.7688.**206**.5

Benoît
de Merly

GUIDA PER L'APPLE 2 . LE ESTENSIONI

E.P.S.I.